MIDAS v1.0

Multimodal Interface for DNA Alignment of Sequences

Bachelor Project (IN3700) 2007

Melissa Cheung, 1228161
Software Technology, Computer Science
Paul van den Haak, 1221760
Media and Knowledge Technology, Computer Science
Philips Innovation Campus, Bangalore, India
University of Technology Delft (Faculty EWI), The Netherlands

Philips supervisor: Dr. N. Dimitrova
TU Delft supervisor: Dr. L.J.M. Rothkrantz
BSc coordinator: Ir. M. Sepers

PHILIPS

TU Delft

Technische Universiteit Delft

Preface

This is the report of the Bachelor Project of two students of Computer Science at the University of Technology Delft in The Netherlands. It is required that all Computer Science students participate in an internship at a company to finalize the Bachelor phase. During this internship software engineering techniques are used to develop a prototype.

For this bachelor project we participated in a three months during internship at Philips Innovation Campus in Bangalore, India. This report describes the software development process of the prototype we developed at Philips. We have developed a prototype that allows Bioinformatics to analyze DNA by sequence alignment and Spectrogram Analysis.

The target groups of this report are software engineers, computer scientists and bioinformatics. The readers of this report should have the basic software engineer knowledge, as this report describes the software development process.

The interesting chapters for the software engineers and computer scientist are chapter 2 till 7, as these chapters contain detailed information about each software development phase. Also a programmer's manual is appended in the appendix. Bioinformatics might be interested in chapter 6.1, which describes how the visualization of the patterns in spectral clustering is implemented, and chapter 8, which describes the developed prototype and its features.

Hereby we would like to thank Dr. Nevenka Dimitrova, Research Fellow and Senior Director, our supervisor at Philips for this project. She has guided us in this project and introduced us to the world of Bioinformatics and Spectral Analysis. Thanks to Dr. Chetan Mittal for explaining about the spectral ordering technique and supporting us in making this prototype suitable to the bioinformatics.

Also we would like to thank Dr. L.J.M. Rothkrantz as our supervisor at the University, for his feedback and support. Thanks to Dr.ir. C.A.P.G. van der Mast for feedback about usablilty. And Dr. P.G. Kluit for helping with our Java design issues.

June 29th 2007,
Melissa Yuen Shan Cheung
Wijnand Paul van den Haak

Summary

Major research efforts in Bioinformatics include sequence alignment, gene finding, genome assembly, protein structure alignment, protein structure prediction, prediction of gene expression and protein-protein interactions, and the modeling of evolution. To perform these specific tasks different tools are used. Using these tools separately is a time consuming, inefficient and expensive process.

MIDAS is a tool that integrates sequence alignment, genome annotation, and spectral clustering and alignment under the same application. The challenge in this project is in representing the knowledge and analyzing the genome data. The DNA data is first transformed into Fourier domain and clustered in MATLAB based on Euclidean distances between the sequences. Our tool allows visualizing the DNA spectra together with a hierarchical tree in a multimodal interface. This in turn enables a bioinformatician to analyze patterns of a group of sequences.

MIDAS is a standalone application which provides an interface around standard sequence alignment tools such as BLAT, ClustalW, as well as newer alignment tools such as Spectrogram analysis via integrating MATLAB code, server connections and data visualizations. JAVA is used as the main programming language during the development of MIDAS. The Spectrogram Analysis script-files in MATLAB are converted into JAVA classes. These classes are used to run standalone MATLAB applications from within JAVA.

Table of Contents

1 Introduction

New developments in the medical world are rising fast. As Philips invests in these new developments, they have a high market share in the medical department. A lot of research projects and concepts developed at Philips will contribute to the future of Medical Systems. One of those projects is "Women's Health: Cross model Bioinformatics for cancer care."

Bioinformatics refers to the creation and advancement of algorithms, computational and statistical techniques, and theory to solve formal and practical problems inspired from the management and analysis of biological data.

Different software tools are used by bioinformatics to analyze DNA. One of those tools is BLAT, an algorithm for sequence alignment, searching large databases of protein or DNA sequences. Another tool is ClustalW, which produces biologically meaningful multiple sequence alignments of divergent sequences. Also a new analyzing approach is the Spectrogram Analysis of genomes. With this technique frequency-domain analysis is done in the genomes using tricolor spectrograms, identifying several types of distinct visual patterns characterizing specific DNA regions. The Spectrogram Analysis is created and developed at Philips by Evan Santo and Nevenka Dimitrova.

Biologically meaningful patterns are found through Spectrogram Analysis, which cannot be found by sequence alignment due to the computational restrictions. When interesting regions and patterns are found, sequence alignment may be required in order to analyze it further.

Currently the sequence alignment is not integrated with the Spectrogram Analysis into one application. This makes searching the genomes a time consuming, inefficient and expensive process.

The aim of this project is:

> *"provide bioinformatics tools that integrate sequence alignment, genome annotation, and spectral clustering and alignment under the same application."*

MIDAS, Multimodal Interface for DNA Alignment of Sequences, will integrate the Spectrogram Analysis, ClustalW and BLAT into one stand-alone application, and extend the application to display genomic annotation as well as several statistics obtained by data mining techniques. The user-interface will cater to the needs of the Bioinformatics researchers.

An iterative and incrementally software development approach is chosen for this project. In this approach the first phases are requirements and planning. During these phases research will be done on bioinformatics and requirements of MIDAS. The next phases are analysis, design and implementation of the MIDAS system. Following, testing the system and evaluation with users will determine the improvements and new requirements for the next iteration.

The structure of this report is based on the software development process. The project definition can be found in chapter 2. In this chapter the aim, background and survey, project environment and plan are discussed. With the project definition defined, the requirements of MIDAS are formed in chapter 3. The next phase is to analyze the system, which is described in chapter 4. In chapter 5 the design of MIDAS is presented. The architectural, technical, java and interface design can be found in this chapter. As follows the implementation phase will be discussed in chapter 6, which explains the visualization of a hierarchical clustered tree, MATLAB functions and problems and solutions of this project. Chapter 7 is dedicated to the testing and evaluation phase, where the usability tests and evaluation is discussed. In chapter 8 MIDAS and its features are presented. The final phase in this project is defining which improvements can be made for future versions, which can be found in chapter 9.

2 Project Definition

2.1 Aim

Provide bioinformatics tools that integrate sequence alignment, genome annotation, and spectral clustering and alignment under the same standalone application.

2.2 Background

In bioinformatics the main topics are data analysis and knowledge representation. In both topics, there are multiple open challenges.

1) Visualization and interactive tools that pull in information from a variety of genomic tools and resources. Currently the computational tools offer only limited visualization capabilities, so there needs to be better information presentation.

2) Enhanced statistical processing with biological knowledge representation. Data analysis and processing currently is dominated with methods that incorporate statistical analysis and pattern recognition.

3) The full analysis however should include an array of tools that will work together to pull in the information from the multiple modalities of information mining: DNA sequence alignment, finding important motifs, and methods for result visualization and clustering. Important aspects cover pulling in information from multiple modalities of measuring processes in gene regulation: transcriptomic data, proteomic data, gene copy number polymorphisms, DNA protein interactions in one unified framework. Currently these modalities are all investigated separately. It can be expected that the problem of multimodality will be tackled in order to solve the problem of causality in molecular events related to gene expression.

2.3 Project description

Currently the sequence alignment tools, like BLAST and ClustalW, are not integrated with the Spectrogram Analysis into one application. This makes searching the genomes a time consuming, inefficient and expensive process.

Our tool has to:

- Integrate the Spectrogram Analysis and sequence alignment tools into one application and extend the application to display genomic annotation as well as several statistics obtained by data mining techniques.
- Make the bioinformatics work less time consuming
- Make the bioinformatics work efficient
- Make the bioinformatics work less expensive
- Be a standalone executable.
- Have a user-interface which will cater to the needs of the Bioinformatics researchers.
- Be tested thoroughly
- Be thoroughly tested during a usability test
- Be supported with a programmer's manual and a user's manual.

The name for this project will be **MIDAS**: **M**ultimodal **I**nterface for **D**NA **A**lignment of **S**equences.

2.4 Project Environment

Project Start Date
April 10, 2007

Project End Date
June 29, 2007

Philips Research Internship Project
Students:
Melissa Yuen Shan Cheung, myscheung@gmail.com
Wijnand Paul van den Haak, paulvandenhaak@gmail.com
University of Technology Delft
Mentor:
Dr. Nevenka Dimitrova, Nevenka.Dimitrova@philips.com

Reliable Care Solutions Department
Philips Research India-Bangalore (PRI-B)
Philips Innovation Campus
MFAR Manyata Tech Park
Manyata Nagar, Nagavara
Bangalore 560 045, INDIA

2.5 Project Plan

The software engineering technique used in this project is iterative and incremental development. In this process the software system is developed incrementally. This allows the developer to take advantage of what was being learned during the development of earlier, incremental, deliverable versions of the system. The advantages come from both the development and use of the system.

The life cycle of iterative and incremental development is displayed in Figure 1.

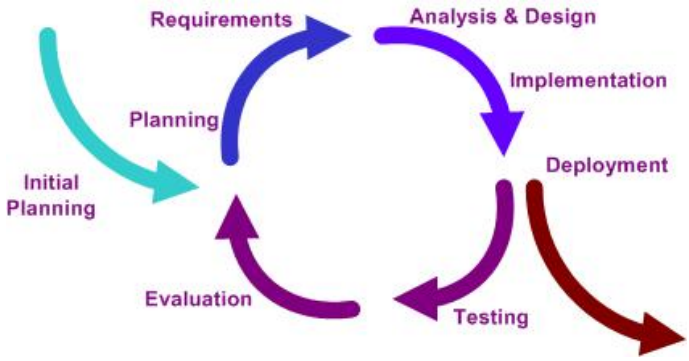


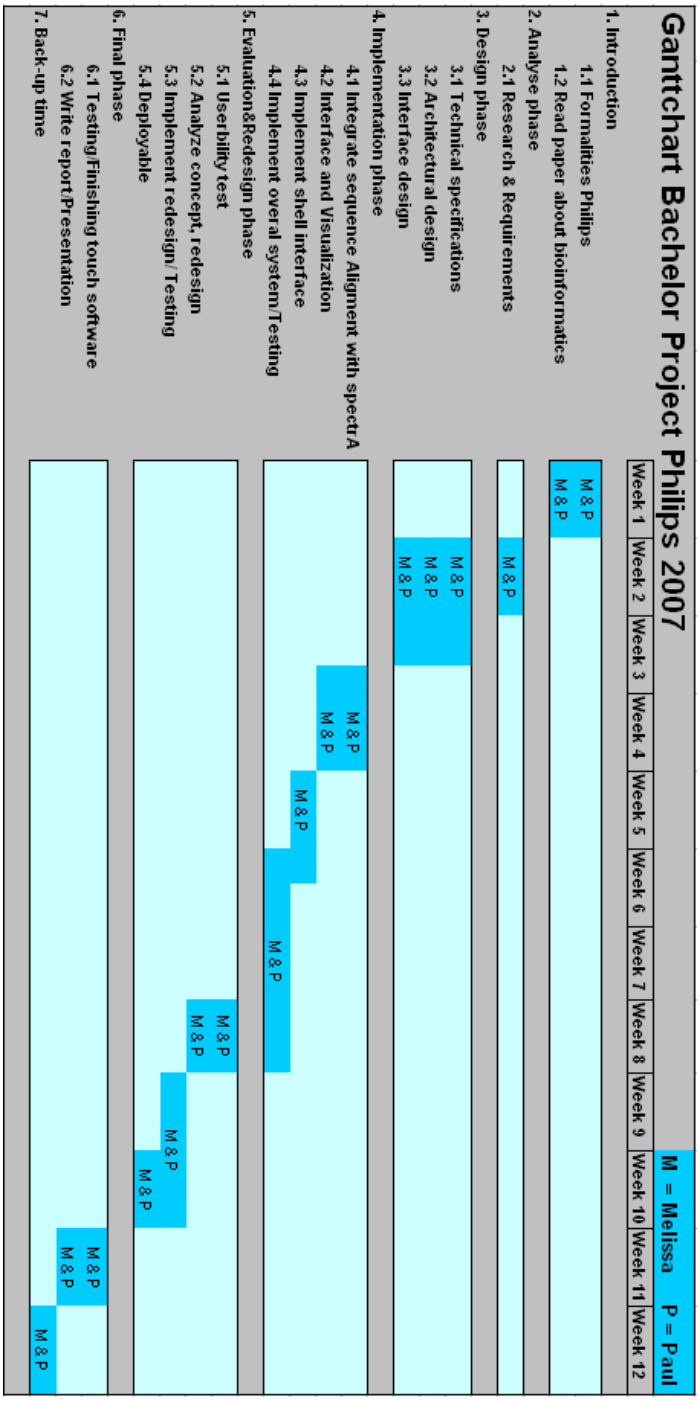
Figure 1 Life-cycle iterative and incremental software development process

A Gantt chart is a chart that illustrates a project schedule. It shows the start and finish dates of the tasks. In this visualization it is easy to get an overall impression on the tasks and deadlines. It is also possible to see the dependencies between the tasks; which tasks are done simultaneously and which are not.

The tasks are divided among twelve weeks as illustrated in figure 2.

The project schedule for MIDAS is illustrated in the Gantt chart (Figure 2).

Figure 2 Gantt chart



3 Requirements

New developments in the medical world are rising fast. As Philips invests in these new developments, they have a high market share in the medical department. A lot of research projects and concepts developed at Philips will contribute to the future of Medical Systems. One of those projects is "Women's Health: Cross model Bioinformatics for cancer care."

This project aims to develop a tool for Bioinformatics. Section 3.1 discusses the current situation. In the Section 3.2, the system description will be presented. This includes the survey, functional and non-functional requirements, constraints and system models. System models will include use case models and object models.

3.1 Current situation

Bioinformatics refers to the creation and advancement of algorithms, computational and statistical techniques, and theory to solve formal and practical problems inspired from the management and analysis of biological data. Major research efforts in the field include sequence alignment, gene finding, genome assembly, protein structure alignment, protein structure prediction, prediction of gene expression and protein-protein interactions, and the modelling of evolution (wikipedia).

Different software tools are used by bioinformatics which perform specific tasks. The best known software is BLAST/BLAT. BLAST/BLAT is an algorithm for sequence alignment, searching large databases of protein or DNA sequences. The NCBI provides a popular web-based implementation that searches their massive sequence databases.

For multiple sequence alignment of DNA or proteins another tool is used; ClustalW. ClustalW produces biologically meaningful multiple sequence alignments of divergent sequences. It calculates the best match for the selected sequences, and lines them up so that the identities, similarities and differences can be seen. Evolutionary relationships can be seen via viewing Cladograms or Phylograms.

Another analyzing approach is the Spectrogram Analysis of genomes. With this technique frequency-domain analysis is done in the genomes using tricolour spectrograms, identifying several types of distinct visual patterns characterizing specific DNA regions. The patterns and their frequency characteristics are related to the sequence characteristics of DNA. Biologically meaningful patterns are found through this method.

3.2 System Description

MIDAS will integrate sequence alignment tools, BLAT and ClustalW, with the Spectrogram Analysis. The application will visualize the output of Spectrogram Analysis, by displaying the genome annotation, spectrogram image and the clustered hierarchical tree. To analyze the output the bioinformaticus (bioinformatician) is able to interactively select genome annotations for (multiple) sequence alignment.

3.2.1 Functional Requirements

User
-Bioinformaticus

Bioinformaticus is able to

Use Spectrogram Analysis

- Load input, FA format file, for Spectrogram Analysis
- Visualize output of Spectrogram Analysis; SpectroVideo, annotation
- Visualize an overview of all the present clustering in the analyzed sequence
- Run Spectrogram Analysis without clustering and without annotation

- Run Spectrogram Analysis without clustering and with full annotation
- Run Spectrogram Analysis with clustering with full annotation
- Run Spectrogram Analysis with clustering without annotation
- obtain the output of Spectrogram Analysis projects in a folder

Philips Linux server
- Login, by giving hostname, username and password
- Connect to the server
- Disconnect to the server

Use BLAST/BLAT
- Load one sequence selected by user from the Spectrogram Analysis annotation list
- Obtain the output of the BLAST/BLAT sequence alignment
- Run BLAT separately

Use ClustalW
- Load one cluster specified by the user from the Spectrogram Analysis by giving the start and end index of that cluster
- Obtain the output of the ClustalW sequence alignment
- Run ClustalW separately

Load external projects
- Load a project which has generate his output on a high-performance grid

Help files
- Open and read help files for MIDAS, BLAT and ClustalW

Save
- Save the output of BLAT or ClustalW as text files

3.2.2 Non-Functional Requirements

Quality requirements
- Performance characteristics
  - System is fast
- Reliable
  - Information, statistics are reliable and correct

Error handling and extreme conditions
- Failure handling
  - Saved data will not be lost

User Interface and human factors
- Clustering, spectrogram image and annotation must be visible on one screen
- ClustalW alignment, partial spectrogram image and annotation must be visible on one screen
- BLAT alignment, partial spectrogram image and annotation must be visible on one screen
- Clusters can be chosen by giving the start and end index of the cluster
- A sequence can be selected as input for BLAT with a mouse click
- Every component should be clearly visible
- Large components should be scrollable

Security issues
- For the connection to the Philips Linux server a Secure Shell protocol must be used

Documentation
- There should be an user-manual
- There should be a programmers-manual
- The code must be well documented

Hardware consideration

- MIDAS can visualize the interface with a minimum resolution of 1280 x 1024 without any loss of information
- A user needs a mouse and a keyboard to give input to MIDAS

General Requirements

- MIDAS must be a deployable standalone executable

3.2.3 Constraints

Platform constraints
Runs on Windows platforms only

Process constraints
This project stands for 420 hours per head.

3.2.4 System Models

In this section the use case diagram, the use cases and the activity diagram are presented.

The use case diagram presents the goals of the bioinformaticus. These goals are described in more detail in the use cases below.

Use case Diagram

Use case Diagram:

Figure 3 Use Case Diagram

Use Cases

Input Chromosome for analysis

Actors:	Bioinformaticus
Pre-conditions:	Bioinformaticus started MIDAS and must have valid Chromosome input (DNA sequence) in FASTA format.
Post-conditions:	After loading the Chromosome input, the system has read this input and additional settings and is ready to run Spectrogram Analysis.
Basic-flow:	1) Bioinformaticus selects Run new Spectrogram Settings 2) Bioinformaticus selects Chromosome Analysis 3) Bioinformaticus inputs valid Chromosome FA file 4) Bioinformaticus sets the required settings 5) Bioinformaticus pushes the OK button in order to run Spectrogram Analysis
Alternative flows:	If the input DNA sequence does not have the FASTA extension or other required settings are not set correctly, an error is given to the bioinformaticus and the bioinformaticus has the possibility to correct his input for the system. If the bioinformaticus pushes cancel button, current windows will close.
Special requirements:	none
Use Case relationships:	includes [Spectrogram Analysis]

Input Discontiguous DNA for analysis

Actors:	Bioinformaticus
Pre-conditions:	Bioinformaticus started MIDAS and must have valid Discontiguous DNA input (DNA sequence) in FASTA format.
Post-conditions:	After loading the Multiple DNA input, the system has read this input and additional settings and is ready to run Spectrogram Analysis.
Basic-flow:	1) Bioinformaticus selects Run new Spectrogram Settings 2) Bioinformaticus selects Discontiguous Analysis 3) Bioinformaticus inputs valid Discontiguous FA file 4) Bioinformaticus sets the required settings 5) Bioinformaticus selects the OK button in order to run Spectrogram Analysis
Alternative flows:	If the input DNA sequence does not have the FASTA extension or other required settings are not set correctly, an error is given to the bioinformaticus and the bioinformaticus has the possibility to correct his input for the system. If bioinformaticus pushes cancel button, current windows will close.
Special requirements:	none
Use Case relationships:	includes [Use Spectrogram Analysis]

Use Spectrogram Analysis

Actors:	Bioinformaticus
Pre-conditions:	Use case [Input Chromosome for analysis] or [Input Discontiguous DNA for analysis] must be completed first.
Post-conditions:	The Spectrogram Analysis has completed
Basic-flow:	1) Bioinformaticus is aware that Spectrogram Analysis is running and sees the progress of analysis in a console output. 2) Bioinformaticus obtains spectrogram images and annotation
Alternative flows:	If the Spectrogram Analysis could not be completed, an error is given to the bioinformaticus in case of an IOException or RuntimeException. In case of a system crash, system must be restarted.

Special requirements: none
Use Case relationships: uses: [Input Chromosome for Analysis]
[Input Discontiguous DNA for analysis]

Create spectrogram video

Actors:	Bioinformaticus
Pre-conditions:	Use case [Use Spectrogram analysis] must be completed first.
Post-conditions:	The spectrogram video has been created.
Basic-flow:	1) Bioinformaticus selects create Spectrogram Video option for a chromosome or a discontiguous chromosome 2) Bioinformaticus is able to watch Spectrogram Video in a MATLAB pop up screen during its creation
Alternative flows:	If the spectrogram video could not be created, an error is given to the bioinformaticus in case of an IOException or RuntimeException.
Special requirements:	none
Use Case relationships:	Includes [Use Spectrogram analysis]

Watch spectrogram video

Actors:	Bioinformaticus
Pre-conditions:	Use case [Use Spectrogram analysis] and [Create spectrogram video] must be completed first.
Post-conditions:	The spectrogram video has been created.
Basic-flow:	1) Bioinformaticus selects 'Open video current tab' option 2) Bioinformaticus is able to watch Spectrogram Video
Alternative flows:	If the spectrogram video could not be shown, an error is given to the bioinformaticus in case of an IOException or RuntimeException.
Special requirements:	none
Use Case relationships:	Includes [Use Spectrogram analysis] and [Create spectrogram video]

Overview all present clustering DNA

Actors:	Bioinformaticus
Pre-conditions:	Use case [Use Spectrogram analysis] must be completed first.
Post-conditions:	The clustering is visible.
Basic-flow:	1) Bioinformaticus selects one spectrogram image 2) Bioinformaticus is able to see clustering in one image
Alternative flows:	If the clustering could not be completed, an error is given to the bioinformaticus in case of an IOException or RuntimeException.
Special requirements:	none
Use Case relationships:	Includes [Use Spectrogram analysis]

Login to Philips Server

Actors:	Bioinformaticus
Pre-conditions:	The server must be online, and user must have a Linux account.
Post-conditions:	Login is authenticated and connection is established.
Basic-flow:	1) Bioinformaticus fills in hostname 2) Bioinformaticus fills in login name 3) Bioinformaticus fills in password 4) Bioinformaticus pushes OK button to login 5) Login is authenticated and connection is established
Alternative flows:	If bioinformaticus pushes cancel button, current windows will close. In case of a system crash, the system has to be restarted. In case of a connection or authentication error, the bioinformaticus will be informed by an error message and he can try again.

Special requirements: None

Use BLAT on output of Spectrogram Analysis

Actors:	Bioinformaticus
Pre-conditions:	Use case [Use Spectrogram Analysis] and [Login to Philips server] must be completed.
Post-conditions:	BLAT output is generated
Basic-flow:	1) Bioinformaticus sets the required and the additional settings 2) Bioinformaticus presses OK button 3) BLAT output is being generated
Alternative flows:	In case of a system crash, the system has to be restarted. In case of a connection error, the bioinformaticus will be informed. If bioinformaticus pushes cancel button, current windows will close.
Special requirements:	The bioinformaticus has to have an account on the Philips server with a login name, password and the IP address of the server.
Use Case relationships:	Includes [Spectrogram analysis] [Login to Philips server]

Use ClustalW on output of Spectrogram Analysis

Actors:	Bioinformaticus
Pre-conditions:	Use case [Use Spectrogram Analysis] and [Login to Philips server] must be completed.
Post-conditions:	ClustalW output is generated
Basic-flow:	1) Bioinformaticus sets the required and the additional settings 2) Bioinformaticus presses OK button 3) ClustalW output is being generated
Alternative flows:	In case of a system crash, the system has to be restarted. In case of a connection error, the bioinformaticus will be informed. If bioinformaticus pushes cancel button, current windows will close.
Special requirements:	The bioinformaticus has to have an account on the Philips server with a login name, password and the IP address of the server.
Use Case relationships:	Includes [Spectrogram analysis] [Login to Philips server]

Load existing project

Actors:	Bioinformaticus
Pre-conditions:	Use case [Use Spectrogram Analysis] must be completed in another session or the output folder has created on another grid.
Post-conditions:	Project is loaded and the spectrogram, annotation and the cluster tree is visible on a tab in the interface
Basic-flow:	1) Bioinformaticus selects the folder where the files are located 2) Bioinformaticus gives the input file which was used to create the output files 3) Bioinformaticus clicks on the OK button
Alternative flows:	In case of a system crash, the system has to be restarted. In case of a connection error, the bioinformaticus will be informed. If bioinformaticus pushes cancel button, current windows will close.
Use Case relationships:	Includes [Spectrogram analysis]

Run BLAT on own input

Actors:	Bioinformaticus
Pre-conditions:	A valid input file with the FASTA (.fa) extension should be present
Post-conditions:	The BLAT output is given to the user on a tab in the interface
Basic-flow:	1) Bioinformaticus sets the required and the additional settings

	2) Bioinformaticus presses OK button 3) BLAT output is being generated
Alternative flows:	In case of a system crash, the system has to be restarted. In case of a connection error, the bioinformaticus will be informed. If bioinformaticus pushes cancel button, current windows will close.

Run ClustalW on own input

Actors:	Bioinformaticus
Pre-conditions:	A valid input file with the FASTA (.fa) extension should be present
Post-conditions:	The ClustaLW output is given to the user on a tab in the interface
Basic-flow:	1) Bioinformaticus sets the required and the additional settings 2) Bioinformaticus presses OK button 3) ClustalW output is being generated
Alternative flows:	In case of a system crash, the system has to be restarted. In case of a connection error, the bioinformaticus will be informed. If bioinformaticus pushes cancel button, current windows will close.

Disconnect from Philips Server

Actors:	Bioinformaticus
Pre-conditions:	Use case [Login to Philips server] must be completed.
Post-conditions:	Bioinformaticus is disconnected from the Philips server
Basic-flow:	1) Bioinformaticus selects Disconnect from server in the menu 2) A pop up is given to the bioinformaticus that the system is disconnected form the Philips server
Alternative flows:	In case of a system crash, the system has to be restarted.
Special requirements:	None
Use Case relationships:	Includes [Login to Philips server]

Object models

An activity diagram represents the overall flow of control of the system. This diagram is presented in figure 4.

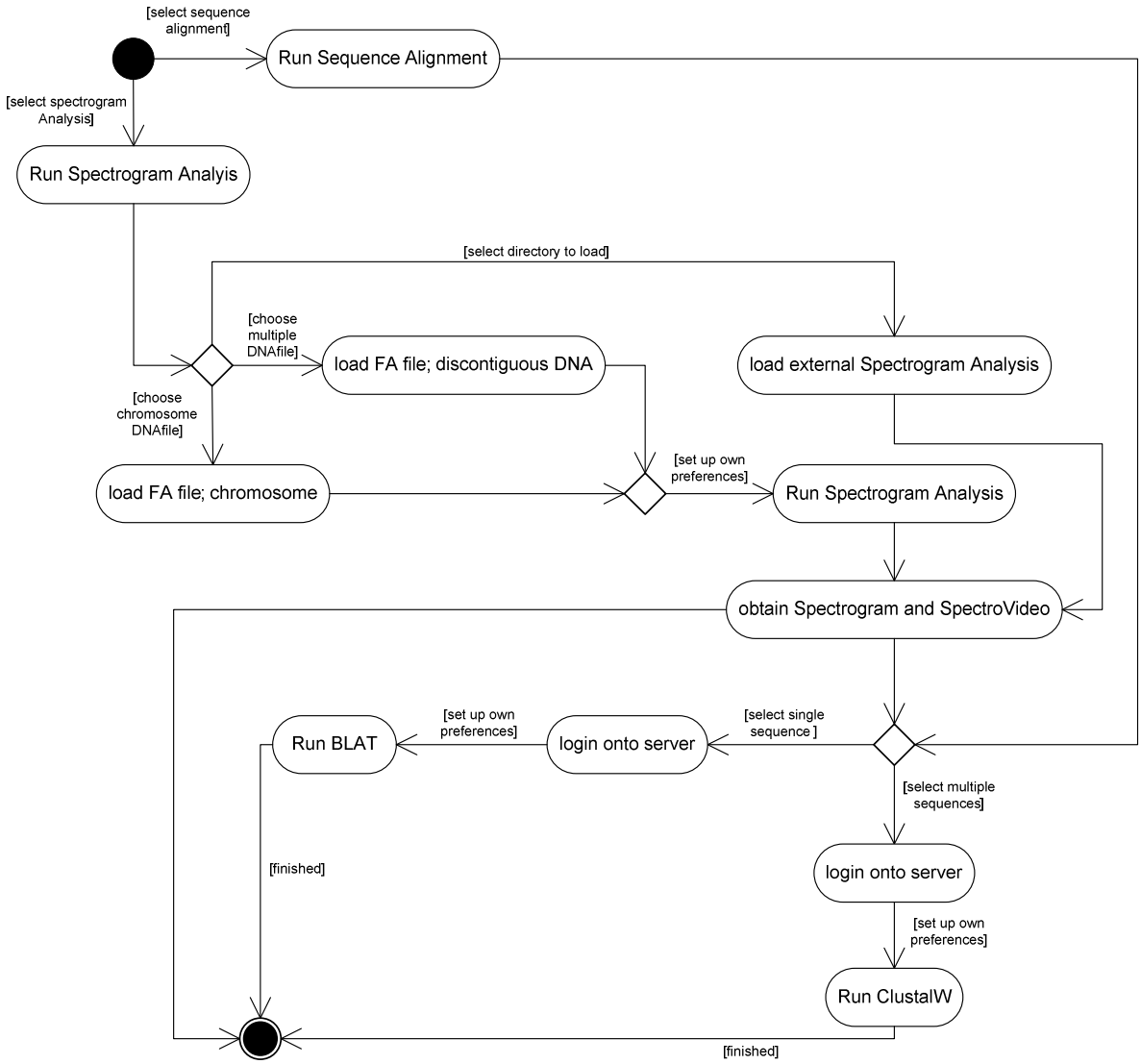


Figure 4 Activity Diagram

4 Analysis

After setting up the requirements for MIDAS, research was done about the preliminary work which is used in MIDAS. This research includes investigation of the Spectrogram Analysis. Also a user analysis for MIDAS is done in the analysis phase.

This chapter describes the preliminary work in Section 4.1. The user analysis in Section 4.2 describes the users profile, presents task diagrams of current and the improved situation and a workflow diagram.

4.1 Preliminary work: Spectrogram Analysis

The preliminary work consists of Spectrogram Analysis, which is created by Evan Santo. Spectrogram Analysis is a powerful tool for analyzing the composition and harmonic properties of genomic sequences.

The first step in Spectrogram Analysis is converting the DNA sequences into binary indicators sequences. The next step is to apply Fourier transforms to obtain the frequency spectrum of each base. The magnitude of a frequency component reveals how strong a certain pattern of the nucleotide base is repeated at that frequency. To improve the readability of the results, each nucleotide base is represented by a color. By combining the frequency spaces of the four bases, a color spectrogram will be presented. This color spectrogram visualizes the compositions and harmonic properties of a sequence.

Spectrogram Analysis will allow for the detection of large and imperfect repetitive elements in the genomes that are very hard to search for by current sequence-space methods. With the Spectrogram Analysis it is possible to systematically analyze the whole genome and analyze cross-species by considering the frequency spectrum.

Hierarchical clustering based on Euclidean distances is applied on the Spectrogram Analysis. The clustering is done because the major drawback of DNA Spectrogram Analysis is the reliance on human perception and memory for the detection of complex patterns. This is impractical for whole chromosomes. Thus clustered spectrogram images allow the extraction of "biological meaning".

Spectrogram Analysis is powerful due to the use of binary indicators instead of nucleotide base symbols. By using binary indicators and converting them into the Fourier domain space, all kinds of signal processing techniques can be used to analyze these signals. Computations on these signals are less heavy than computations on long nucleotide base symbols. Thus Spectrogram Analysis is able to analyze very long sequences, up to whole genomes.

(References:
N. Dimitrova, Y.H Cheung, M.Q. Zhang, Analysis and Visualization of DNA Spectrograms: Open Possibilities for the Bioinformatics Research, ACM MM 2006, Oct 25, 2006, pp. 1017 - 1024
N. Dimitrova, E. Santo. Improvement of spectral analysis as a genomic analysis tool)

4.2 User Analysis

In the analysis phase it is important to understand the user of your product. Getting a complete understanding of your user will contribute to the quality of your end product. As a first step in a user analysis, it is important to form a user profile, Section 4.2.1. After forming this profile the user was analyzed in the current situation (Section 4.2.2) and in the improved situation (Section 4.2.3), the situation after project MIDAS is completed. Section 4.2.4 contains the workflow diagram which presents the workflow in the new, improved situation.

4.2.1 User profile

A well formed user profile will contribute to the success of MIDAS. When the developer has a deep understanding of the psychological characteristics, knowledge and experience, function and task characteristics and physical characteristics of the user, he can fit the product to their needs. The user profile is described below.

Psychological characteristics:	
Attitude	interested in information about genes, DNA
Motivation	he will be happy using MIDAS because it gives him the possibility to speed up their work during analysis of genes
Knowledge and experience:	
Reading experience:	high
Type experience	high
Education	high
System experience	average
Application experience	average
Use of other systems	average
Computer knowledge	high
Programming skills	average
Language	English, Hindi
Function and task characteristics:	
Frequency of use	daily
Training	is important because it will help the user in first stage using MIDAS
Use of system	bioinformatics
Priority	MIDAS is important for the user
Task structure	tasks will decrease using MIDAS
Physical characteristics:	
Color blindness	possible
Sex	man/woman
Age	25+

4.2.2 Current situation

Analyzing a single chromosome or a discontiguous chromosome is a time consuming job currently. A lot of tasks have to be done manually by each employee of Philips. Task diagrams give a clear view on the tasks which an employee of Philips performs to analyze a chromosome in the current situation.

Figure 5 displays the task diagram for analyzing a discontiguous chromosome and Figure 6 displays the task diagram for analyzing a single chromosome.

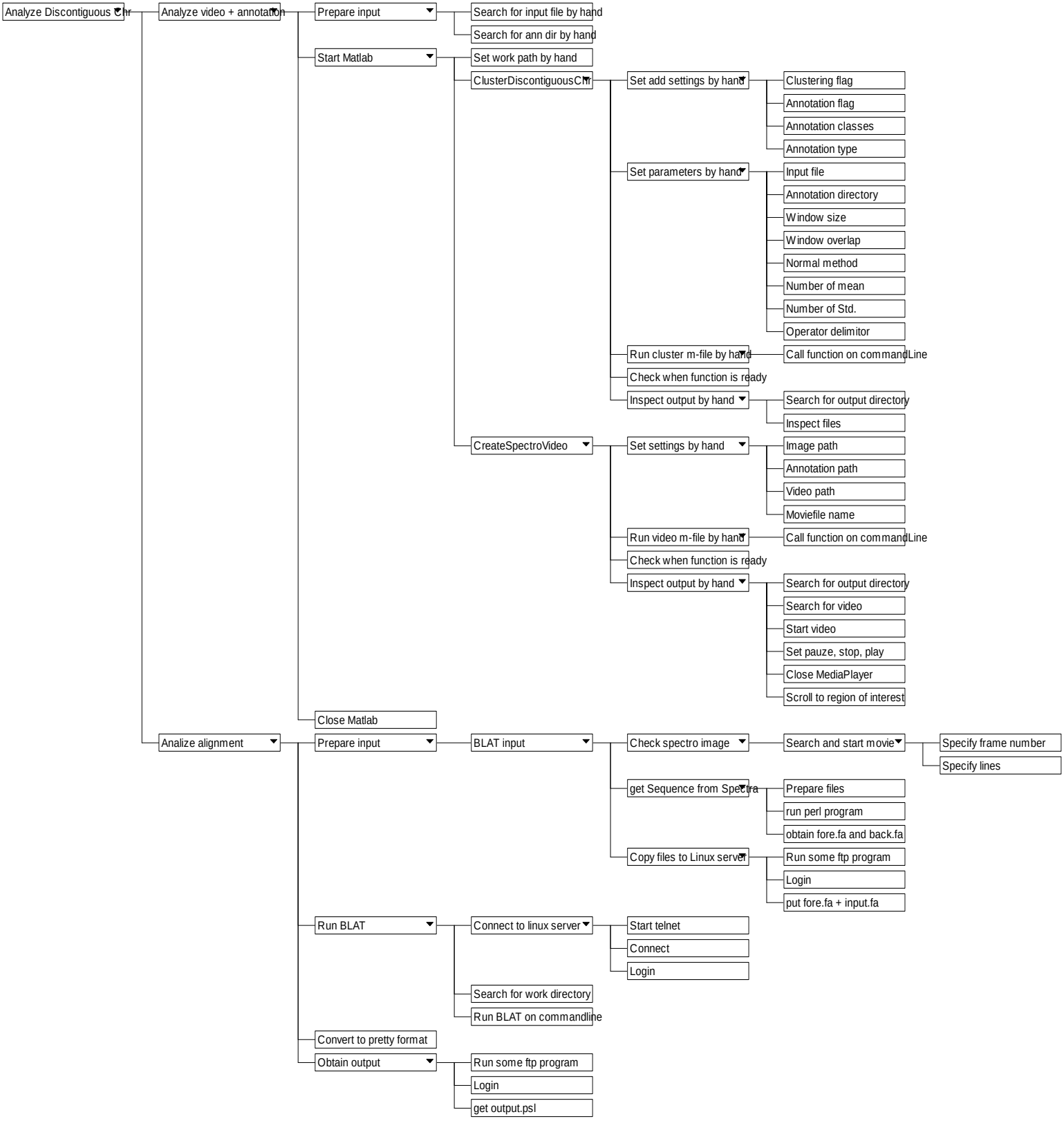


Figure 5 Task diagram discontiguous analysis

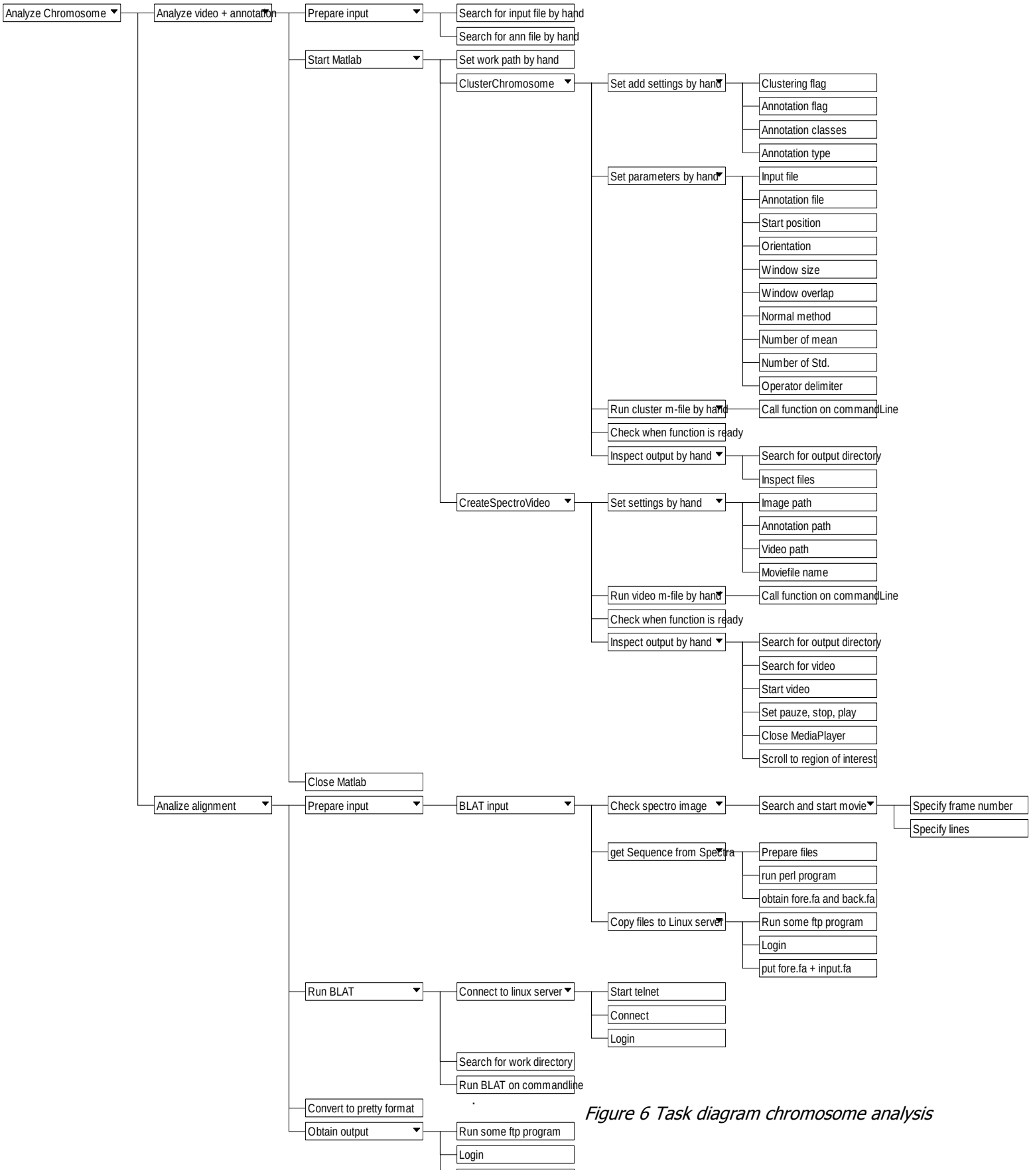


Figure 6 Task diagram chromosome analysis

Each block in the diagram is one task. By examining all the tasks, it becomes clear that a lot of tasks can be automated. The user has to hardcode input paths, output folders, standard settings, etc. Then the user runs the script-files manually on the command line in MATLAB. This approach is sensitive for errors and the MATLAB command line does not accept a single type error.

The user has to stay focused during the whole process; when a function call is executed, he has to execute the next one with the right parameters. This approach cannot be called user-friendly.

4.2.3 Improved situation

The development of MIDAS will contribute to the efficiency during analysis of DNA. MIDAS provides an interface where most of the tasks in the current situation are automated or simplified. The user does not have to give complete function calls to the command line or set all the parameters manually. MIDAS performs this task.

Figure 7 gives an impression of the tasks which must be performed by the user in the improved situation by analyzing a discontiguous chromosome. Figure 8 displays analyzing a single chromosome.

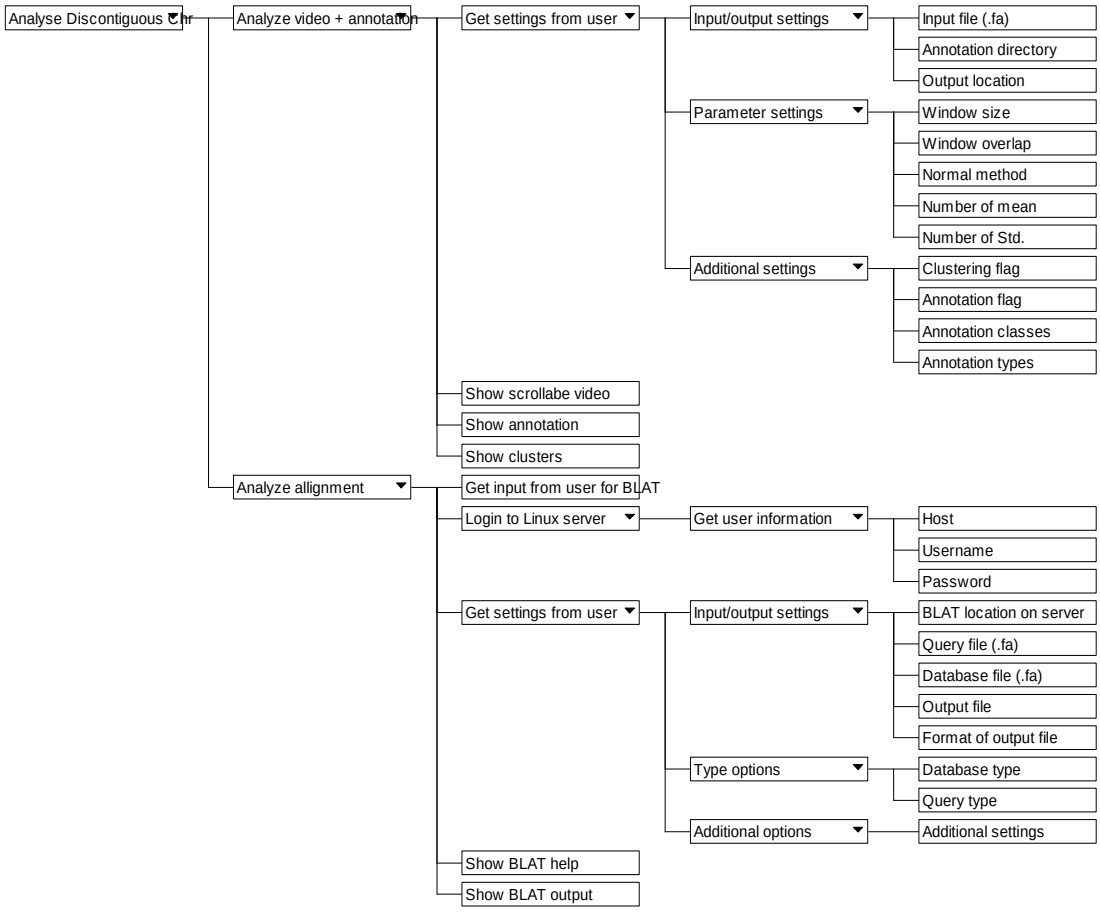


Figure 7 Improved task diagram, analyzing discontiguous chromosome

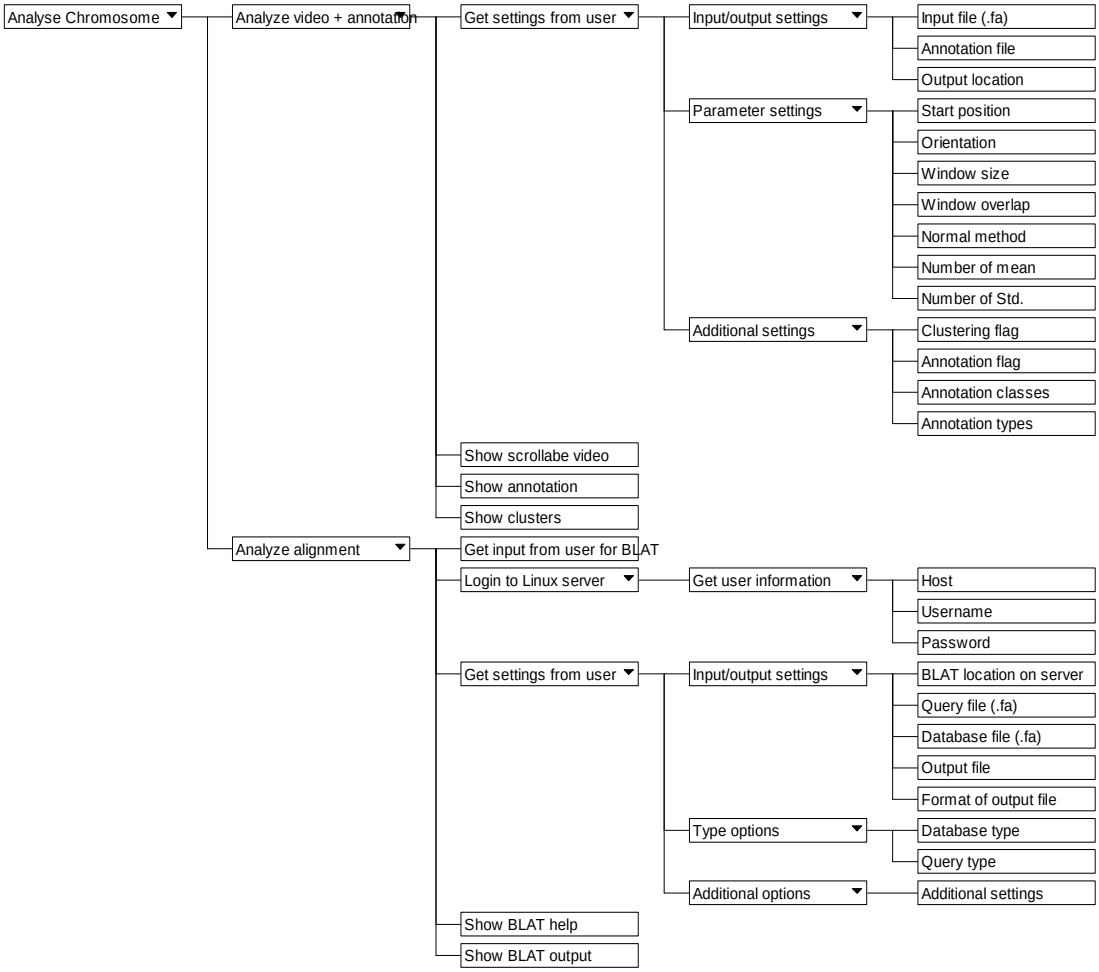


Figure 8 Improved task diagram, analyzing chromosome

Comparing the tasks diagrams of the current situation with the tasks diagrams of the improved situation it becomes clear that a lot of tasks are automated or not necessary anymore. This increases the efficiency of analyzing DNA.

4.2.4 Workflow

A workflow diagram (Figure 9) is designed during the analysis phase to visualize the operational part of the work procedure. Since the task diagrams of the improved situation are simpler and shorter compared to the task diagrams of the current situation, the operations in this workflow diagram are separated into manual tasks and automatic tasks to visualize this difference.

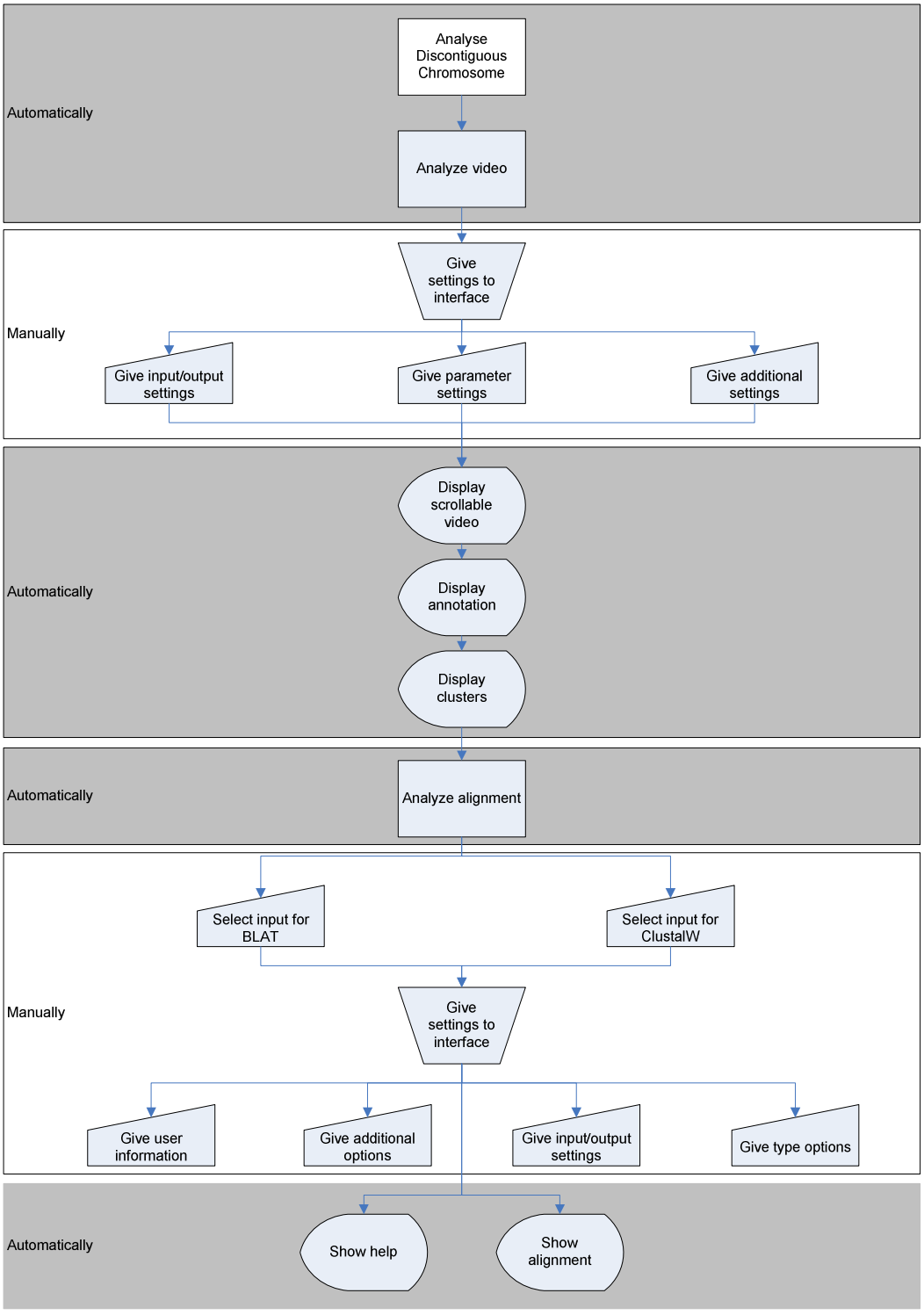


Figure 9 Workflow Diagram

5 Design

This chapter describes the Design phase of this project. The architectural design is presented in 5.1. This outlines the design of architectural components of MIDAS. As follows the technical design and specification is presented in 5.2. In this Section the techniques used in this project are discussed. In 5.3 the Java design is presented with class diagrams and sequence diagrams. In 5.4 the interface design issues and storyboards can be found.

5.1 Architectural design

The architectural design describes the possibilities of the system. This chapter presents the design goals, present outline of the system and subsystems and the structure of the system.

5.1.1 General priorities - design goals

The next goals are the guidelines for the design process:

- Efficient
- Low maintenance
- User-friendly
- Reliable
- Flexible
- Stand-alone

5.1.2 Outline of the design

The aim of this project is to develop a tool which integrates Spectrogram Analysis and sequence alignment, and provides visualization of patterns in the spectrograms.

The Spectrogram Analysis is preliminary work executed by Evan Santo. This program consists of MATLAB scripts. The sequence alignment tool BLAT is originally a web-based application, as it aligns a sequence with the genome, and the genome database are accessible by Internet. ClustalW is a tool which aligns multiple sequences. ClustalW can be installed on Windows, Linux and MacOS. For this project the sequence alignment tools and genome database, are accessible from an internal Philips Linux Server.

The system contains the following layers (Figure 10):

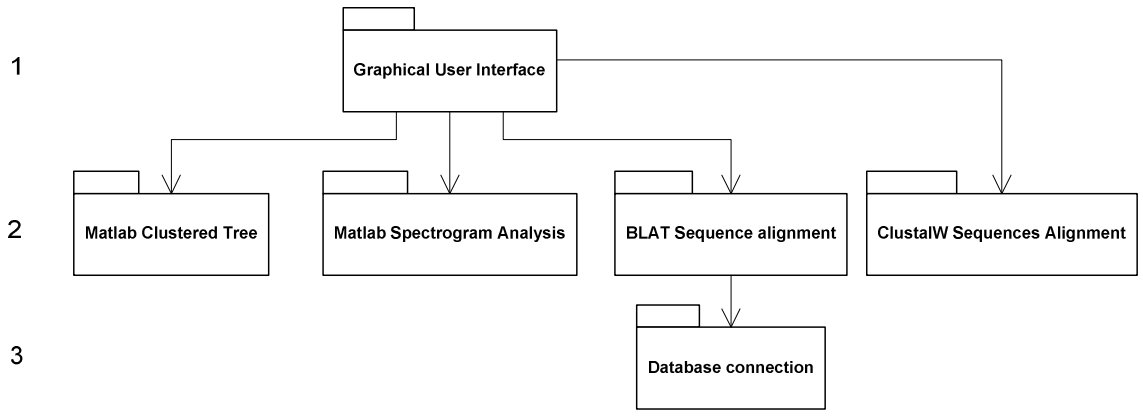


Figure 10 Layer overview

1. The Graphical User Interface is a user friendly environment to control the application. The user is able to run Spectrogram Analysis and sequence alignment and see a visualization of the patterns of the spectrogram.

2. This layer contains all the mathematical functionality.
3. The BLAT program connects to a database to perform the sequence analysis.

5.1.3 Major design issues

During the conceptual phase, two major issues came up. The first issue was to decide in which programming language the Graphical User Interface should be implemented. After deciding on the programming language, the second issue was were to put the main control of the program.

Graphical User Interface programming language

The major issue for designing and implementing the Graphical User Interface is deciding which programming language is the best option. To decide on the programming language aspects like performance, client-server connections, error handling, platform dependencies, user-friendly interface and web possibilities are considered. This leads to the issue:

"What is the programming language of the Graphical User Interface for integrating spectrogram Analysis, BLAT and ClustalW application?"

The options for programming language are:

- MATLAB
- C++
- Java
- C#

MATLAB could be considered as the easiest option, because the Spectrogram Analysis is written in MATLAB. MATLAB has toolbox named Guide which provides Graphical User Interface building. Disadvantage is that the toolbox is limited in its features. Also, previous work experience with this toolbox showed that action handlers in MATLAB are not easy to work with. As MIDAS is a comprehensive system, which needs a user interface that can handle complex action handlers, using MATLAB for GUIs is considered as a bad option. This narrows the options to C++, Java and C#.

This projects programmers' background lies in object oriented programming. All the options support object-oriented programming. Both Java and C# are object oriented languages and as C++ is a statically typed free-form multi-paradigm language, it also supports object oriented programming. Considering the time scope of this project and the programmer's background, C++ is not the best option. Although C++ is almost unlimited in programming possibilities, it is a powerful but complex language. Learning C++ and using all its resources would consume too much time in this project.

There are many similarities between Java and C#. However, there are also many differences, with C# being described as a hybrid of C++ and Java, with additional new features and changes. Both C# and Java offer a broad range of resources for designing GUIs and action handlers. Differences can be found in the exception handling. Java supports checked exception, while C# only supports unchecked exception. Checked exceptions enforce the programmer to declare all exceptions thrown in a method, and to catch all exceptions thrown by a method invocation. This is more consistent in error handling. C# is a relatively new language with a smaller user base than Java. This is due to the fact that Java is older and Java's network-computing support; the Java applets are often used on the web.

Java is the best option for MIDAS. The exception handling is very useful as MIDAS is a rather complex system. Also the larger user base, history, and the large amount of useful open source packages Java offers can contribute to the performance and consistency challenges of

MIDAS. Of course, integrating three different tools into one application might be troublesome in consistency. The network-computational support is very convenient for MIDAS as it has to communicate with the Philips Linux Server. And in the future, MIDAS might become a web application.

Main control of MIDAS

The following issue is:
"How to combine the mathematical computations of MATLAB as well as the sequence alignment tools with the Graphical User Interface in Java?"

There are 2 possible options:

Option 1 (Figure 11)

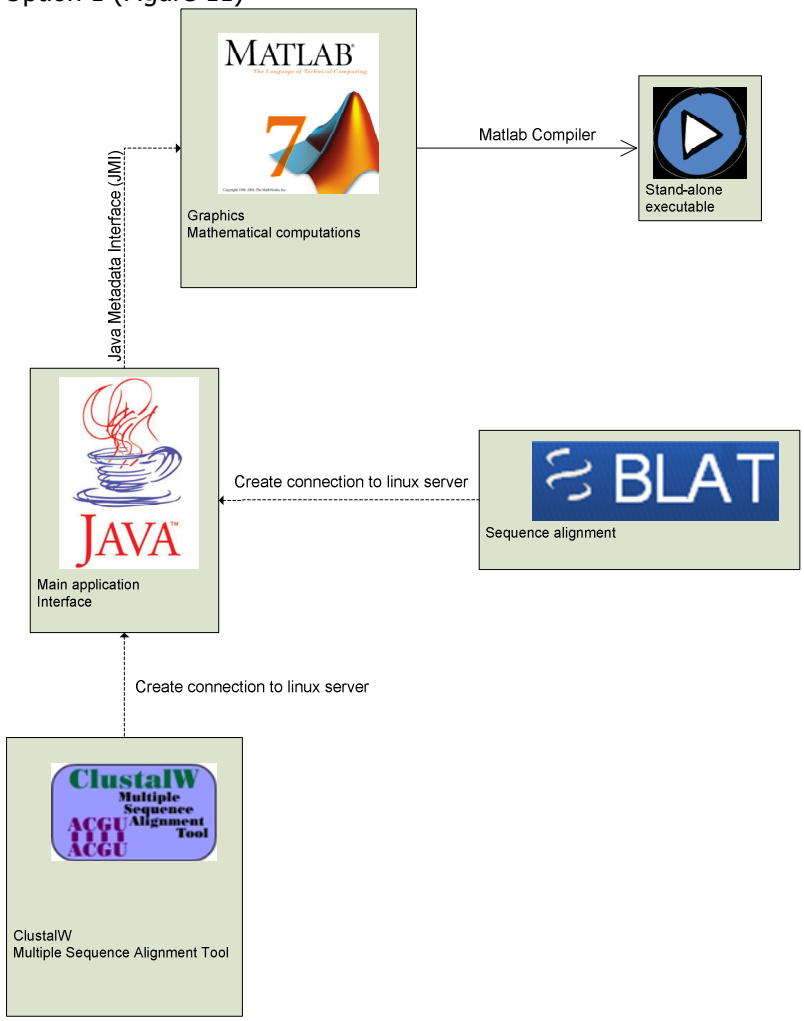


Figure 11 System overview

In option 1 the application will be run as a stand-alone executable MATLAB program. Thus all the source code has to be integrated in MATLAB. MATLAB contains a Java Virtual machine (JVM) and Java Metadata Interface (JMI) which will run the java classes. The GUI is created in Java and will make a connection to a Linux server where BLAT and ClustalW are installed. The output of BLAT and ClustalW will be displayed in the GUI. MATLAB will produce the computations and images of the Spectrogram Analysis, and visualize it in the GUI.

Option 2 (Figure 12 System Overview)

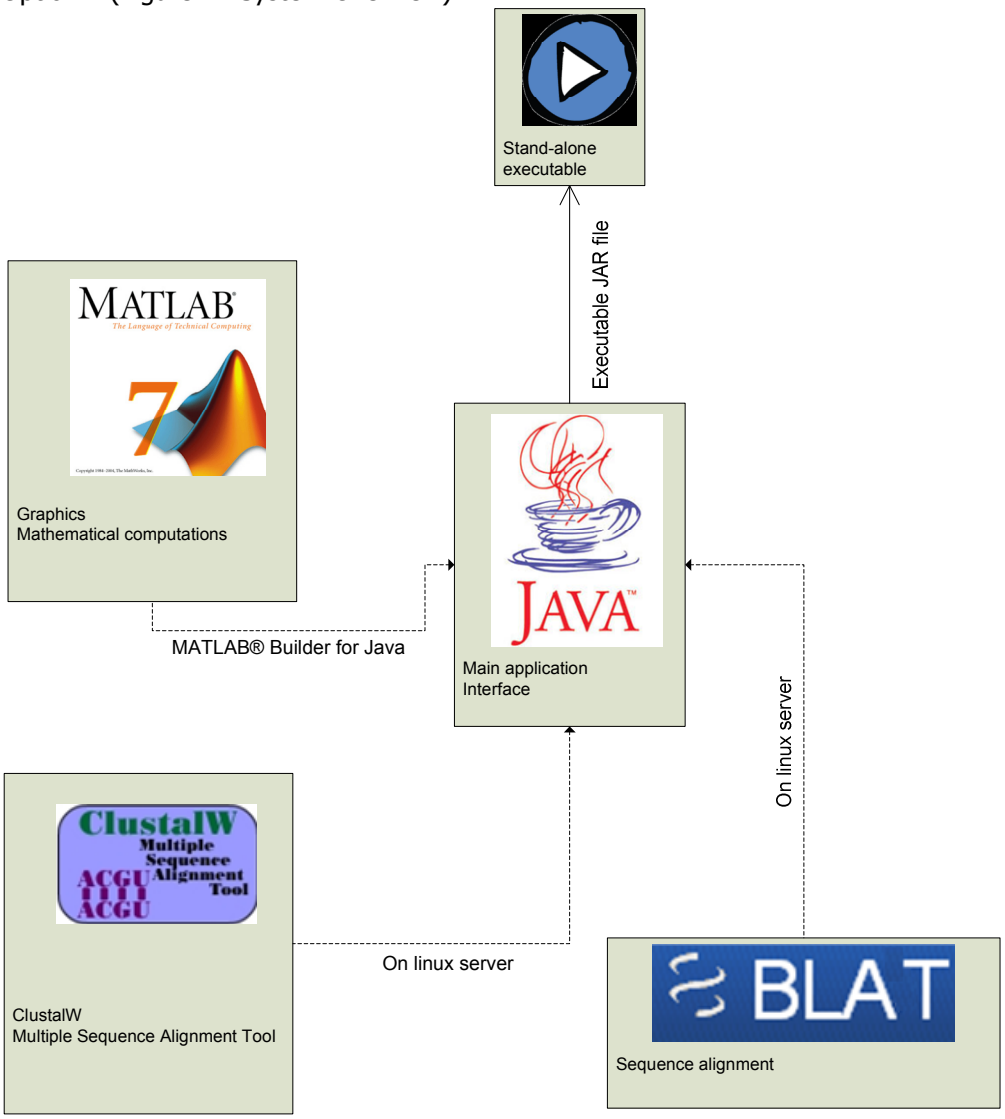


Figure 12 System Overview

In option 2 the application will be run as a stand-alone executable Java program. Thus all the source code has to be integrated in Java. One approach is to rewrite all the MATLAB scripts into Java classes. This approach is not a suitable solution. Rewriting all the files is a time consuming process. Also by rewriting the scripts into Java code, the computational power of the high level matrix optimized language will be lost. Thus this leads to the other approach, which is to build the MATLAB scripts into Java classes with the MATLAB Builder for Java. The converted MATLAB scripts run on a MATLAB Component Runtime, which is a stand-alone set of shared libraries required for executing MATLAB, based components on computers without MATLAB. These converted scripts can be called in Java, and the GUI visualizes the output. The main application creates a connection to a Linux server to get the output of BLAT and ClustalW, and displays this in de GUI.

Option 2 is the solution for this issue. Both options were investigated on several criteria to meet the design goals. The most important ones are efficiency and performance.

The first option is to integrate Java in MATLAB and the main application will be a MATLAB executable. Although it is possible to integrate Java into MATLAB, two disadvantages are found.

- Integrating Java into MATLAB is not as easy as it seems. It is a time consuming process.
- MATLAB is an interpreted language, thus slower than Java.

Next to the disadvantages of the integration of the two languages, there were some other issues.

- The GUI is in Java, where the programs should be linked to each other. It seems illogical to make a MATLAB executable.
- Creating MATLAB executables requires the MATLAB Compiler 4.5. The issue is that MATLAB Compiler does not support Java Objects, thus compiling the application as stand-alone is not possible.

The second option uses new MATLAB Deployment Target: MATLAB Builder for Java 1.0. This is an extension of the MATLAB Compiler 4.5. This tool converts MATLAB algorithms into standard Java classes. The new builder eliminates the time-consuming and error-prone process of recoding an algorithm created in MATLAB into Java. Disadvantage is that the new software has to be purchased. Also, it is necessary to know how to use the MATLAB tools.

Considering these two options, the second one is the only solution, because it is not possible to create a stand-alone application of MATLAB integrated Java files in MATLAB.

5.2 Technical design and specification

This Section describes the technical design and specifications of MIDAS. In Section 5.2.1 the technologies used for MIDAS are described: software, libraries and used plug-ins. Next in Section 5.2.2 the component diagram and the deployment diagram are described in order to clarify the MIDAS components in its environment.

5.2.1 Software

The programming environments are Java and MATLAB. Java is used for the User Interface, integrating the programs and for the connection to the Linux server. MATLAB is used for its computational functionality for the data mining part of this project. For programming in Java and MATLAB, the next software was used.

Eclipse 3.2 - Java
MATLAB R2006b - MATLAB

MIDAS runs BLAT, ClustalW and Spectrogram Analysis under one application. The specifications of these programs are given below.

BLAT
BLAT is a software tool for bioinformatics which performs rapid mRNA/DNA and cross-species protein alignments. BLAT is more accurate and 500 times faster than popular existing tools for mRNA/DNA alignments and 50 times faster for protein alignments at sensitivity settings typically used when comparing vertebrate sequences.
(http://www.genomeblat.com/)

ClustalW
ClustalW is a general purpose multiple sequence alignment program for DNA or proteins. It produces biologically meaningful multiple sequence alignments of divergent sequences. It calculates the best match for the selected sequences, and lines them up so that the identities, similarities and differences can be seen. Evolutionary relationships can be seen via viewing Cladograms or Phylograms. (http://www.ebi.ac.uk/clustalw/)

Spectrogram Analysis
Spectrogram Analysis is developed in MATLAB at Philips Innovation Campus by Evan Santo. This is a program, composed of different executable script files, which produces a Spectrogram Analysis of genomes and also outputs a Spectrogram Video. With this technique frequency-domain analysis is done in the genomes using tricolor spectrograms, identifying several types of distinct visual patterns characterizing specific DNA regions. The patterns and their frequency characteristics are related to the sequence characteristics of DNA. Biological meaningful patterns are found through this method.

For MIDAS, the BLAT and ClustalW software are located on an intern Linux server. This is to prevent inaccessibility due to invisibility of the Internet and because it runs faster on an intern server. Thus the application needs to communicate with the server in order to run BLAT or ClustalW. Spectrogram Analysis was rewritten to make it compatible for conversion into Java classes. The conversion of MATLAB code into Java classes is nowadays easier with a new tool of MATLAB, called MATLAB Builder for Java. Below we discuss all the plug-ins, libraries and tools which are needed.

MATLAB Compiler 4.6
MATLAB Compiler lets you automatically convert your own MATLAB programs into self-contained applications and software components and share them with end users. Applications and components created using MATLAB Compiler do not require MATLAB anymore to run. (www.mathworks.com)

MATLAB Builder for Java 1
MATLAB Builder for Java extends the MATLAB Compiler with tools for automatically generating Java™ classes from your MATLAB algorithms. You can run MATLAB based classes created by using Builder for Java outside the MATLAB environment, referencing them the same way as any other Java class. (www.mathworks.com)

Ganymed-ssh2
Ganymed-SSH-2 for Java is a library which implements the SSH-2 protocol in pure Java (tested on J2SE 1.4.2 and 5.0). It allows one to connect to SSH servers from within Java programs. It supports SSH sessions (remote command execution and shell access), local and remote port forwarding, local stream forwarding, X11 forwarding, SCP and SFTP. There are no dependencies on any JCE provider, as all crypto functionality is included. (http://www.ganymed.ethz.ch/ssh2/)
License: BSD licenses (the Berkeley Software Distribution license). (Appendix A)

The Graphical User Interface of MIDAS relies on the Java JFC package. JFC is short for Java Foundation Classes, which encompass a group of features for building graphical user interfaces (GUIs) and adding rich graphics functionality and interactivity to Java applications. For MIDAS the Swing toolkit is used, which is a part of the JFC.
(http://java.sun.com/j2se/1.5.0/docs/api/javax/swing/package-summary.html)

A stand-alone application is only complete with help files. A software tool is used to create a windows based help file (.hlp).

Shalom Help Maker v0.6.1
Shalom Help Maker is a Windows help file editor (.hlp). SHM is your editor and you can produce the help file with SHM and the enclosed compiler.
(http://www.fileflash.com/program/2348/)
License: Freeware

In order to make MIDAS a stand-alone application, more software tools were used. One of these tools is the FAT-jar, a special deployment tool for Eclipse. Another tool is used to create a windows installer. As follows the description of these software tools are presented.

FAT-Jar
The Fat Jar Eclipse Plug-In is a Deployment-Tool which deploys an Eclipse java-project into one executable jar. In addition to the eclipse standard jar-exporter referenced classes and jars are included to the "Fat-Jar", so the resulting jar contains all needed classes and can be executed directly with "java -jar", no class path has to be set, no additional jars have to be deployed. Jars, External-Jars, User-Libraries, System-Libraries, Classes-Folders and Project-Exports are considered by the plug-in. The Main-Class can be selected and Manifest-files are merged. The One-JAR option integrates a specialized Class-Loader written by Simon Tuffs (http://one-jar.sourceforge.net/) which handles jar-files inside a jar.
Individual files and folders can be excluded or added to the jar.
(http://fjep.sourceforge.net/)
License: GNU General Public License (GNU GPL/GPL). (Appendix B)

Setup Editor v2.1.0.33
Setup will create very small Windows installations easily and quickly without all the overhead of the big tools.
Perfect for emailing a few files to your friends or distributing very small applications without the end user having to stress over how to install and run them.
It has a full script editor to make your life even easier, a command to create Windows shortcuts and a tool to quickly create self extracting zip files. Optional compression is now fully integrated.
(http://www.glenn.delahoy.com/software/index.shtml)
License: Freeware

5.2.2 Environment and Component Models

To illustrate and clarify the relations between the components of MIDAS in their execution environment, the component diagram and deployment diagram of MIDAS are created.

The component diagram's main purpose is to show the structural relationships between the components of a system. In Figure 13 the MIDAS component is related to all the other components. The user commands the Graphical User Interfaces of the Spectrogram Analysis, Hierarchical Clustered Tree, ClustalW and BLAT. He can use these GUIs to input the arguments he prefers. The outputs of these computations are visualized in the MIDAS component.

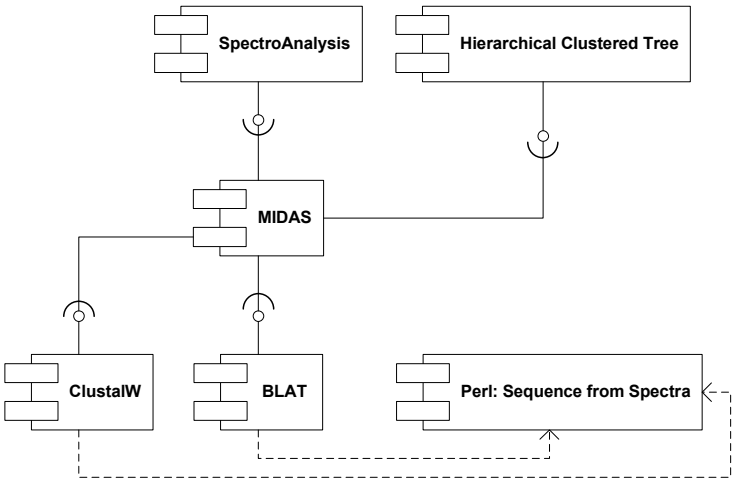


Figure 13 Component Diagram

The deployment diagram (Figure 14) illustrates the components in their execution environment. This diagram shows the hardware for your system, the software that is installed on that hardware, and the middleware used to connect the disparate machines to one another. The deployable MATLAB Component Runtime, converted Spectrogram Analysis MATLAB package and the Main Application, MIDAS, are located on the client machine. The main application MIDAS connects to the Philips server. The sequence alignment tools, BLAT and ClustalW, are located on this server. The Perl program to fetch the sequences from the Spectrogram Analysis in order to run sequence alignment is executed on the server, which is a Linux environment. It is also possible to load a Spectrogram Analysis in MIDAS; this would be the case when the Spectrogram Analysis is computationally too heavy to handle for the client machine. The Spectrogram Analysis can be executed in MATLAB on a high performance platform and loaded into MIDAS.

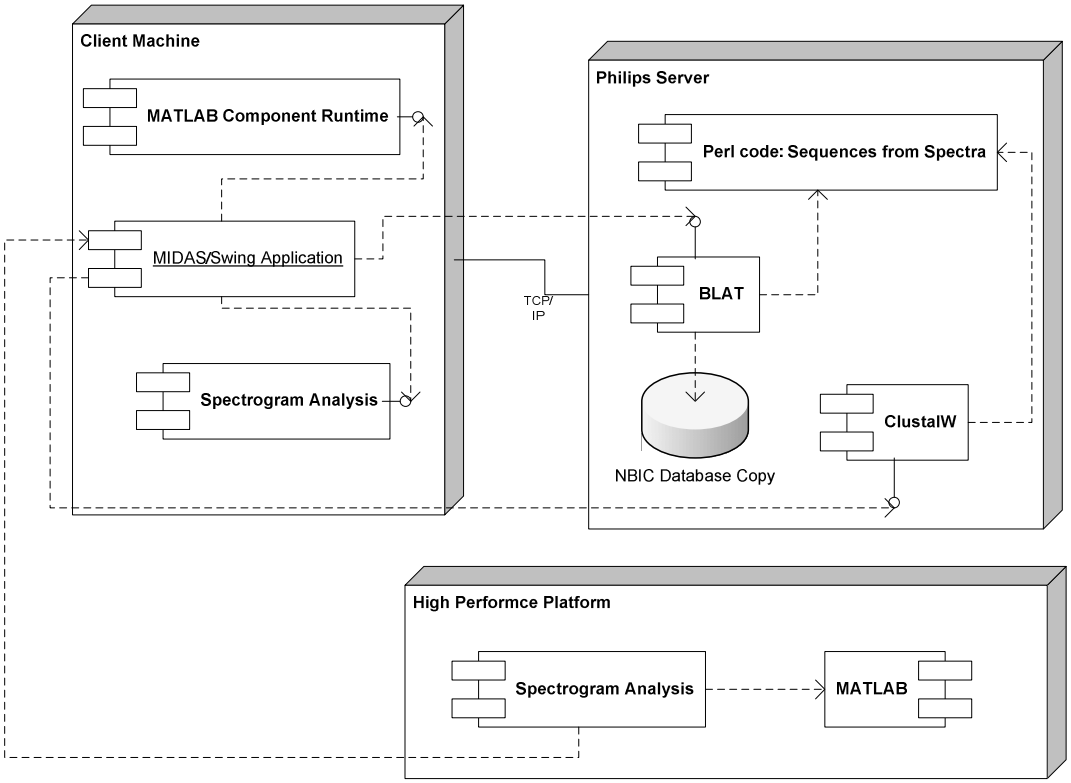


Figure 14 Deployment Diagram

5.3 Java Design

To design the Java structure of MIDAS the Unified Modeling Language is used. Several models have been created, as seen in the Technical design and specification. In this chapter the class diagram is presented and the class descriptions. Because MIDAS is a broad system, with many classes containing many methods, the attributes and methods are left out.

5.3.1 Class Diagram

In Figure 15 the class diagram is presented. There are four packages; MIDAS, spectrogramAnalysis, sequenceAlignment and tools.

MIDAS

This package contains the main application, thus the main function to run MIDAS. The outputs of the Spectrogram Analysis and sequence alignment are combined in this package. One of the most important objects is the General Settings object (GeneralSettings class). This general settings object contains all the important information which needs be known and can be used in the whole system. The classes for visualization of the hierarchical clustered tree are also in this package.

Spectrogram Analysis

This package contains the GUIs for input of the Spectrogram Analysis, and the "call" to run Spectrogram Analysis. The converted MATLAB functions to Java methods for Spectrogram Analysis are also in this package.

sequence alignment

This package contains the client-server classes to connect to the Philips Server. Also the GUIs and the execution commands for BLAT and ClustalW are in this package.

Tools

This package contains "tools" which can be used in all the classes. The console for the systems processes is one tool. Another tool is the file chooser, a directory/file browser for selecting files and directories. Another tool is to save a text file of some output on your computer.

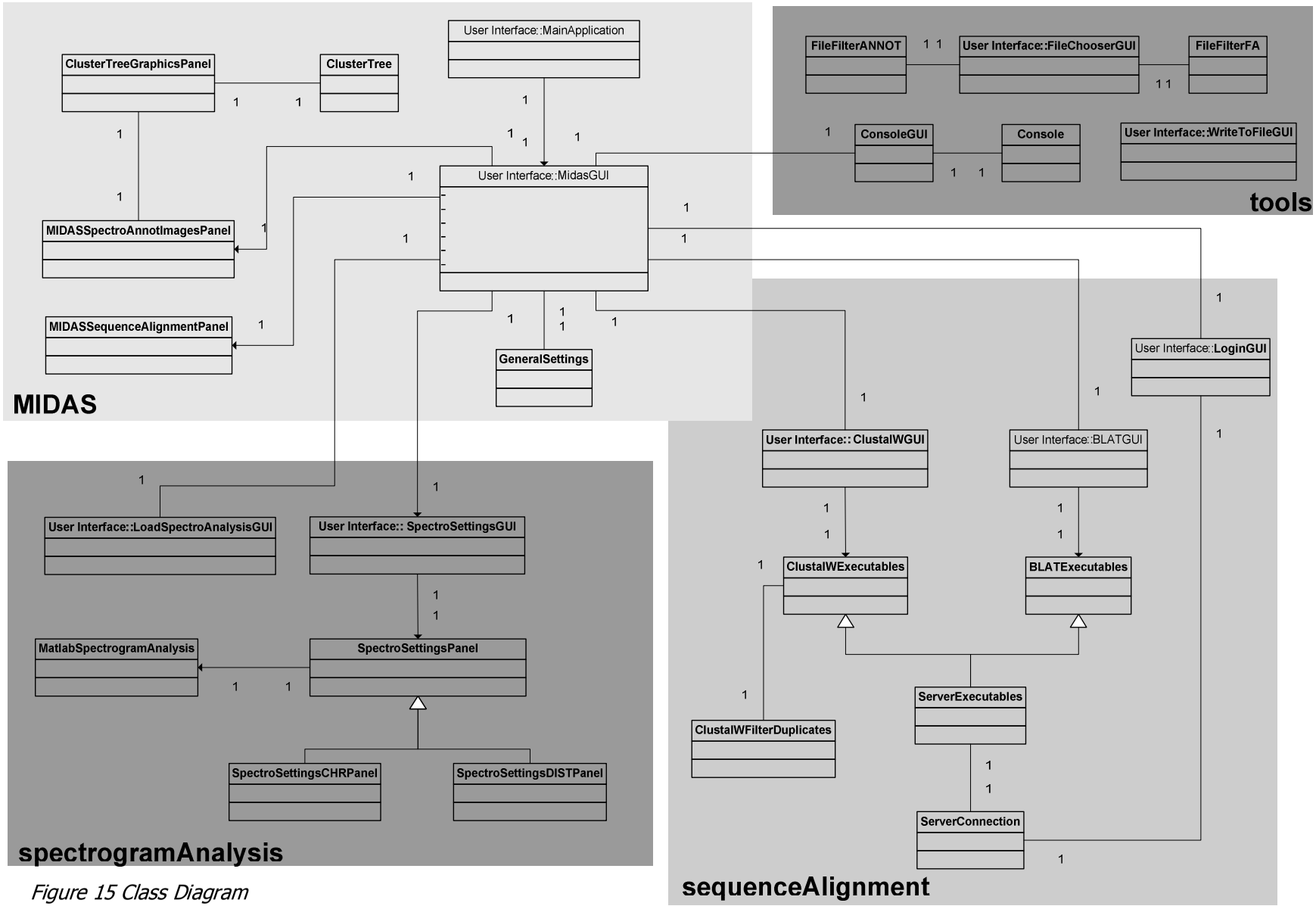


Figure 15 Class Diagram

5.3.2 Detailed class diagram

The description of all the classes is given in this section.

MIDAS

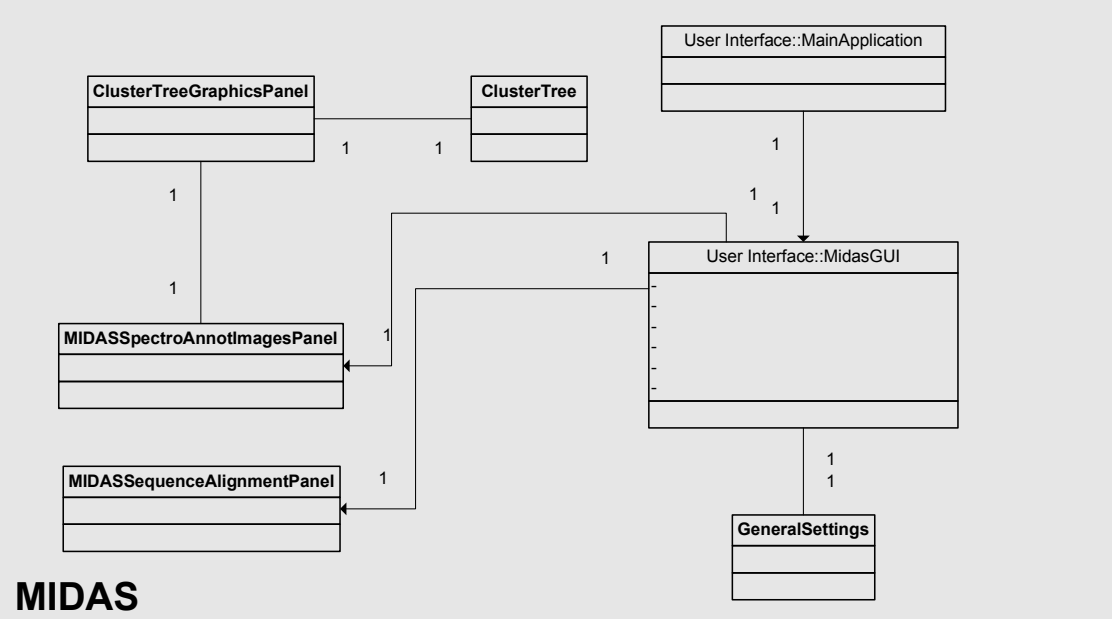


Figure 16 MIDAS package

GeneralSettings

GeneralSettings is an object which can be given to all the classes in the MIDAS system, which contains all the important information that is needed to run parts of the system. These settings are for example output directories, input values for sequence alignment, location of files, etc.

MainApplication

MainApplication initializes MIDAS.

MIDAS

MIDAS is main frame in which all sub frames are initialized. The main control is in this class. This class also initializes MATLAB converted code for spectrogram video.
MIDAS contains two nested classes for layout.
-TabCloseIcon for close button tabs and closing the tabs
-BackgroundPanel for the background image on a panel

MIDAS is restricted to run one process (either sequence alignment or Spectrogram Analysis) at the time for consistency and performance. This is handled by disabling the MIDAS interface for the user during a running process.

MIDASSequenceAlignmentPanel

MIDASSequenceAlignmentPanel is the panel for the output of sequence alignment. This loads BLAT output or ClustalW output. Depending on the mode, user input (mode 1) or from a Spectrogram Analysis (mode 0), it loads the correct output. This panel contains a MATLAB converted function, the original file is: CreatePartialImage.m

MIDASSpectroAnnotImagesPanel

MIDASSpectroAnnotImagesPanel is the panel for output of Spectrogram Analysis, contains spectrogram images, slider to go through the images, the annotation and the cluster tree. Additional is the sequence alignment panel to select annotation from the annotation overview to run on BLAT or to give a range to run on ClustalW.

NOTE: The sequence alignment panel is now only available for Discontiguous SpectrA, as the code for extracting the sequences from SpectrA only supports Discontiguous sequences.

ClusterTree

ClusterTree builds a Hierarchical tree of Clusters. It reads the file created by MATLAB buildClusterInfo.m, findClusterCorrespondingToPerm.m. Then it sorts the information into hashmaps and computes the coordinates needed to draw a clustered Hierarchical Tree.

NOTE: If one sequence has more than MAX_RECURSION_COUNTER clusters, the branch for this will sequence will not be drawn.

CLusterTreeGraphicsPanel

ClusterTreeGraphicsPanel is the panel with the hierarchical tree. The component paints the tree according to the coordinates computed by ClusterTree. The Spectrogram Images corresponding with this tree all have a fixed dimension with a maximum size of 505x750. Thus the panel is fixed to the height of 750. The tree is drawn for each range; the coordinates are translated in order to display the correct part of the tree.

NOTE: If one sequence has more than MAX_RECURSION_COUNTER clusters, the branch for this will sequence will not be drawn.

spectrogramAnalysis

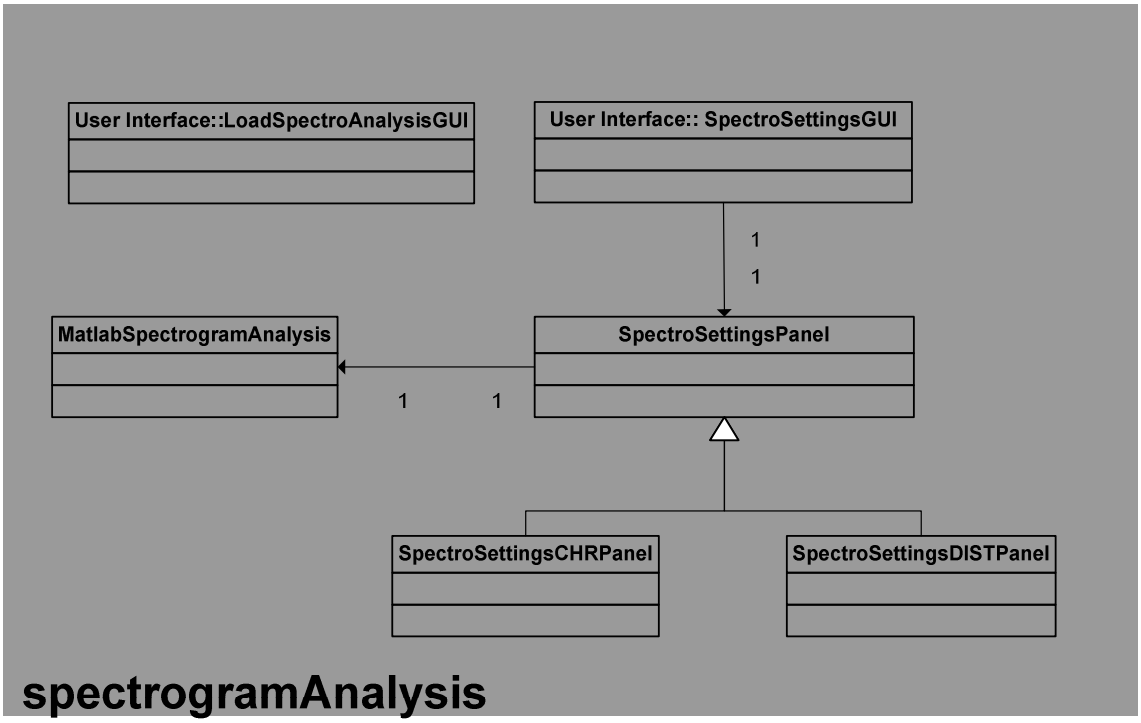


Figure 17 Spectrogram Analysis package

MATLABSpectrogramAnalysis

MATLABSpectro contains the Java methods for the Spectrogram Analysis MATLAB scripts. These scripts were made by Evan Santo. The methods are build to JAVA by the MATLAB JAVA BUILDER 1.0. The converted scripts are:

- ClusterDistcontiguousSpectrogram.m
- ClusterChromosomeSpectrogram.m
- createChromosomeClusteredSpectogram.m
- createDistcontiguousClusteredSpectogram.m
- createClusteredVideo.m
- createSpectroImages.m

For the detailed documentation about this code please refer to the MATLAB code.

SpectroSettingsGUI

SpectroSettingsGUI is the GUI for the Spectrogram Analysis Settings Panel. This class initializes the SpectroSettingsCHRPanel and SpectroSettingsDISTPanel.

SpectroSettingsPanel

SpectroSettingsPanel contains the general Spectrogram settings for Distcontiguous and Chromosome Clustering.

SpectroSettingsCHRPanel

SpectroSettingsCHRPanel is the panel of chromosome Spectrogram Analysis. This class extends SpectrogramSettingsPanel. This panel contains all the settings to run a chromosome Spectrogram Analysis.

SpectroSettingDISTPanel

SpectroSettingsDISTPanel is the panel of discontiguous Spectrogram Analysis. This class extends SpectrogramSettingsPanel. This panel contains all the settings to run a discontiguous Spectrogram Analysis.

LoadsSpectroAnalysisGUI

LoadsSpectroAnalysisGUI is the interface for loading of an existing Spectrogram Analysis directory. This class loads a Spectrogram Analysis project. The directory should contain several files and directories in order to work.

Files:

- Annotation.txt
- AnnotationFULL.txt (not necessary)
- Fourier.txt
- Image.txt

Directories with files:

- Annotation: Annotation.mat
- Clustering: Distance.mat, Linkage.mat, Permutations.mat
- FreqMatrix: FreqMatrix.mat
- Sequence: Either sequenceDiscontiguous.mat or sequenceChromosome.mat
- SpectroImages: All spectrogram images should be in this folder

sequenceAlignment

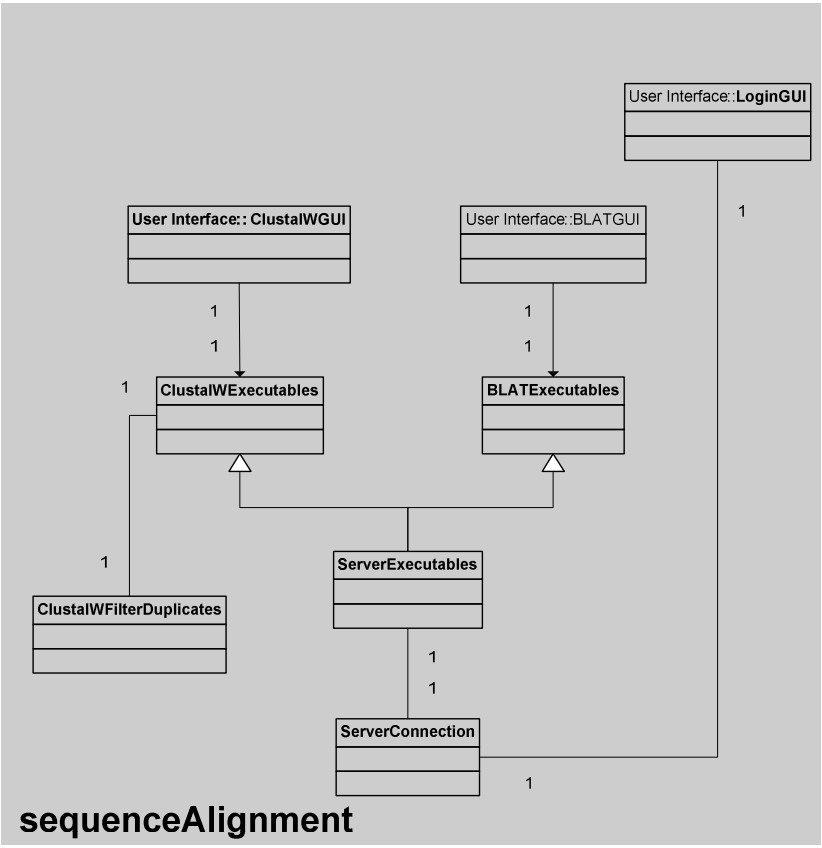


Figure 18 Sequence alignment package

ServerConnection

ServerConnection connects to the Philips server and authenticates. It uses ganymed-ssh2 for JAVA for SSH connection.

LoginGUI

LoginGUI is the interface for the server connection. The connection is created in ServerConnection.

ServerExecutables

SeverExecutables contains server commands, file synchronization methods. Uses ganymed-ssh2 for JAVA for a SSH connection.

ClustalWExecutables

ClustalWExecutables passes the commands to the server. ClustalWExecutables extends ServerExecutables. This class gives the run CustalW command, and puts files on and retrieves from the server.

ClustalWGUI

ClustalWGUI is the User Interface for ClustalW. This GUI uses ClustalWExecutables for running ClustalW and synchronizes files with the LINUX Server. On this GUI additional settings can be set for ClustalW.

ClustalWFilterDuplicates

ClustalWFilterDuplicates filters the duplicate sequences for ClustalW. This class reads a file and writes a file with the duplicates removed if needed.

BLATExecutables

BLATExecutables passes the commands to the server. BLATExecutables extends ServerExecutables. This class gives the run BLAT command, and puts files on and retrieves from the server.

BLATGUI

BLATGUI is the User Interface for ClustalW. This GUI uses BLATExecutables for running ClustalW and synchronizes files with the LINUX Server. On this GUI additional settings can be set for BLAT.

Tools

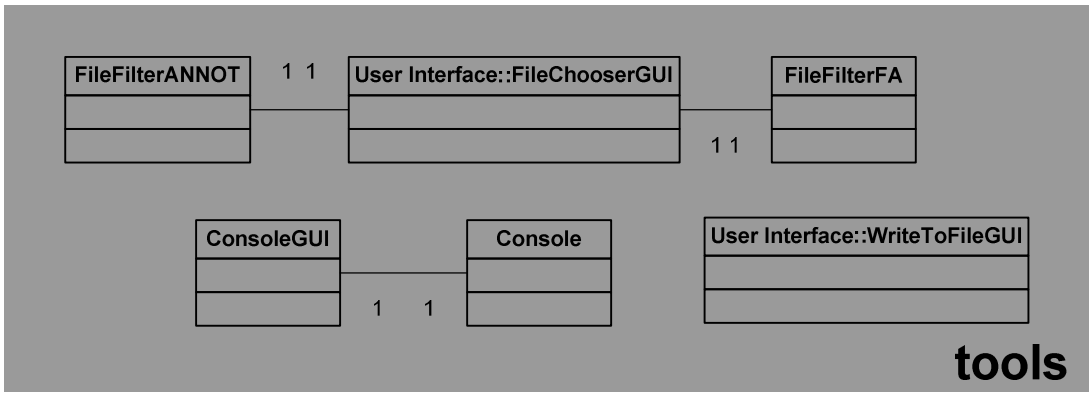


Figure 19 Tools

WriteToFileGUI

WriteToFileGUI is an interface which allows a user to set an output directory and the name of the file to be written from the output obtained in MIDAS. This is used for BLAT and ClustalW outputs.

Console

Console replaces the JAVA console. Console displays the JAVA console output.

ConsoleGUI

ConsoleGUI is the interface of the Console. ConsoleGUI contains a text area which prints the stacktrace and displays a progress bar.

FileChooserGUI

FileChooserGUI is a user interface for browsing. jFileChooser(java.sun.com) is used for browsing through own directories. This class contains methods to obtain a pathname to load, or save the files needed for input/output for MIDAS

FileFilterAnnot

FilterFileANNOT extends FileFilter to filter the correct format for input of MIDAS. This filter filters .annot files.

FileFilterFA

FilterFileFA extends FileFilter to filter the correct format for input of MIDAS. This filter filters .fa files.

5.3.3 Sequence Diagrams

Sequence diagrams illustrate the flow the system by depicting the interactions between the objects. As MIDAS is a comprehensive system which has a great deal of features, many sequence diagrams can be produced. In this Section only the most important sequence diagrams are presented; the flows which are not that obvious. The main functionality of MIDAS is to run a Spectrogram Analysis and provide a visualization of the output, and from the output the user has to be able to select annotations for sequence alignment. Sequence diagrams are created for this main functionality. This functionality has been split up in three sequence diagrams.

The first sequence diagram (Figure 20) illustrates the interactions between objects when the user runs a Spectrogram Analysis. This sequence diagram describes the choices the user has, analyzing a chromosome or discontiguous, and illustrates the flow of running a discontiguous Spectrogram Analysis.

The second (Figure 21) and third (Figure 22) are sequence diagrams of running sequence alignment in BLAT and ClustalW respectively. In the sequence diagrams of run BLAT and run ClustalW, two objects were left out, due to space and the clarity of the sequence diagram. These two objects are:

- LoginGUI uses the class ServerConnection for connection to the server
- BLATExecutables extends the class ServerExecutables
- ClustalWExecutables extends the class ServerExecutables

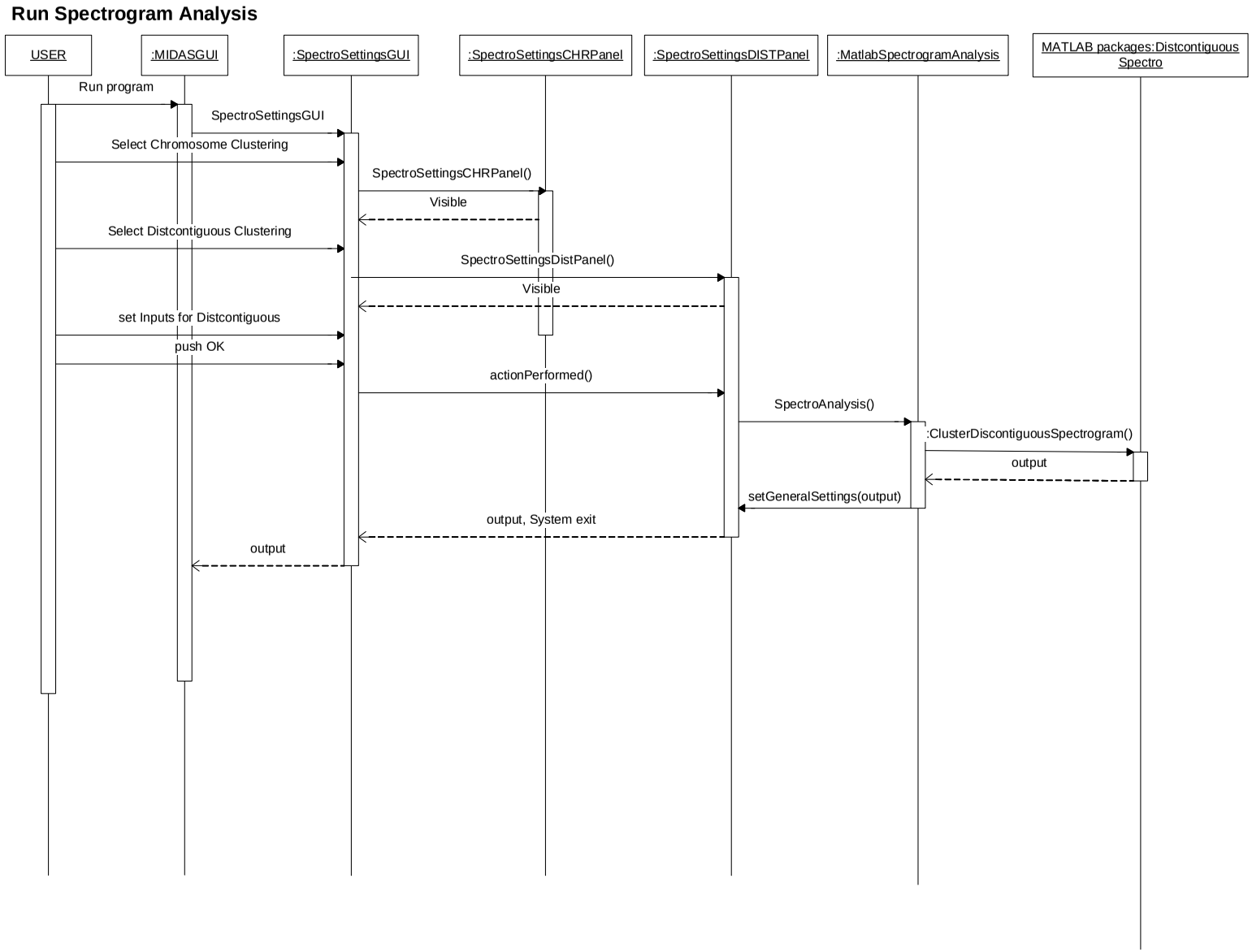


Figure 20 Sequence Analysis Sequence Diagram

Run BLAT

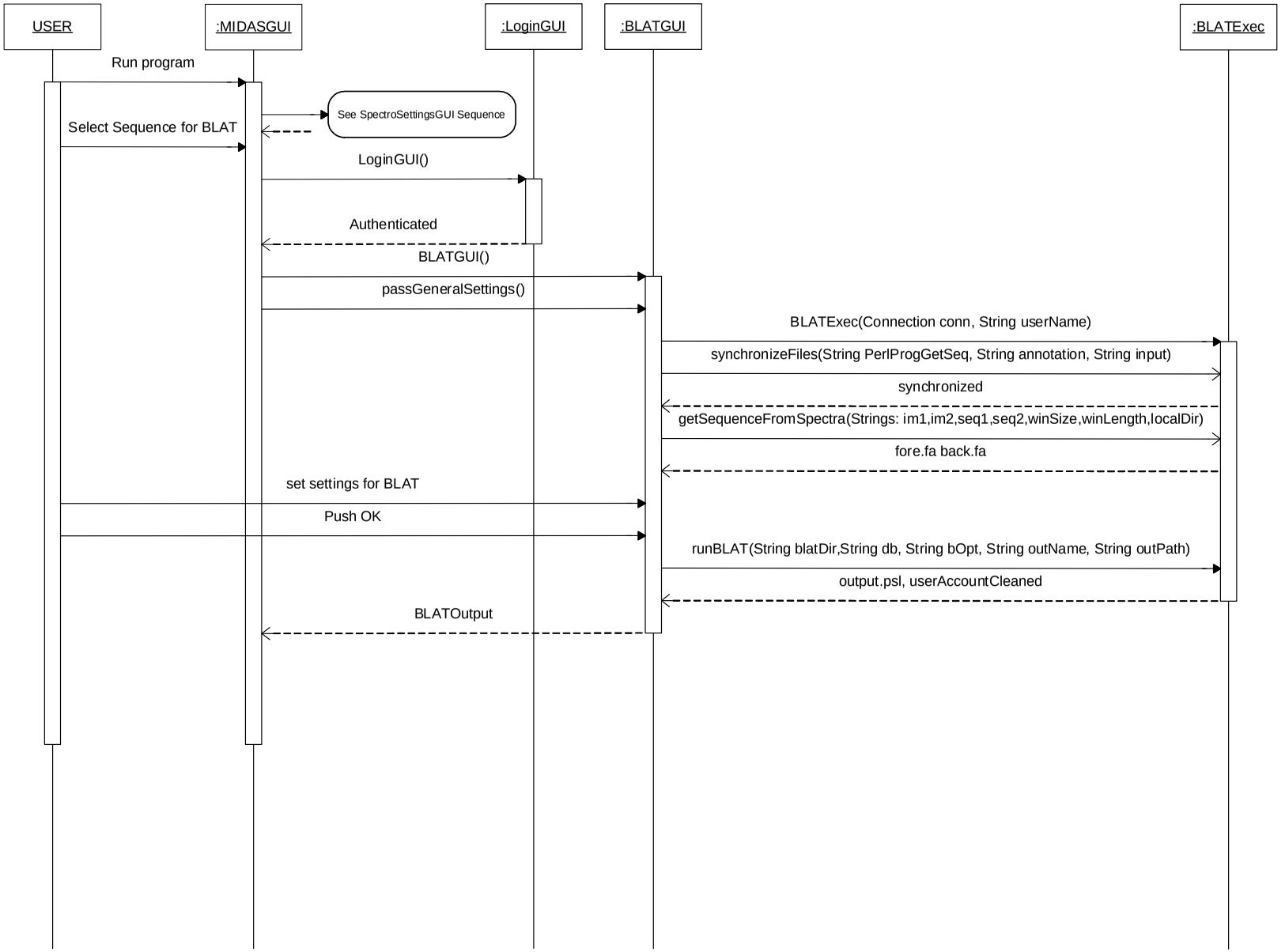


Figure 21 BLAT Sequence Diagram

Run ClustalW

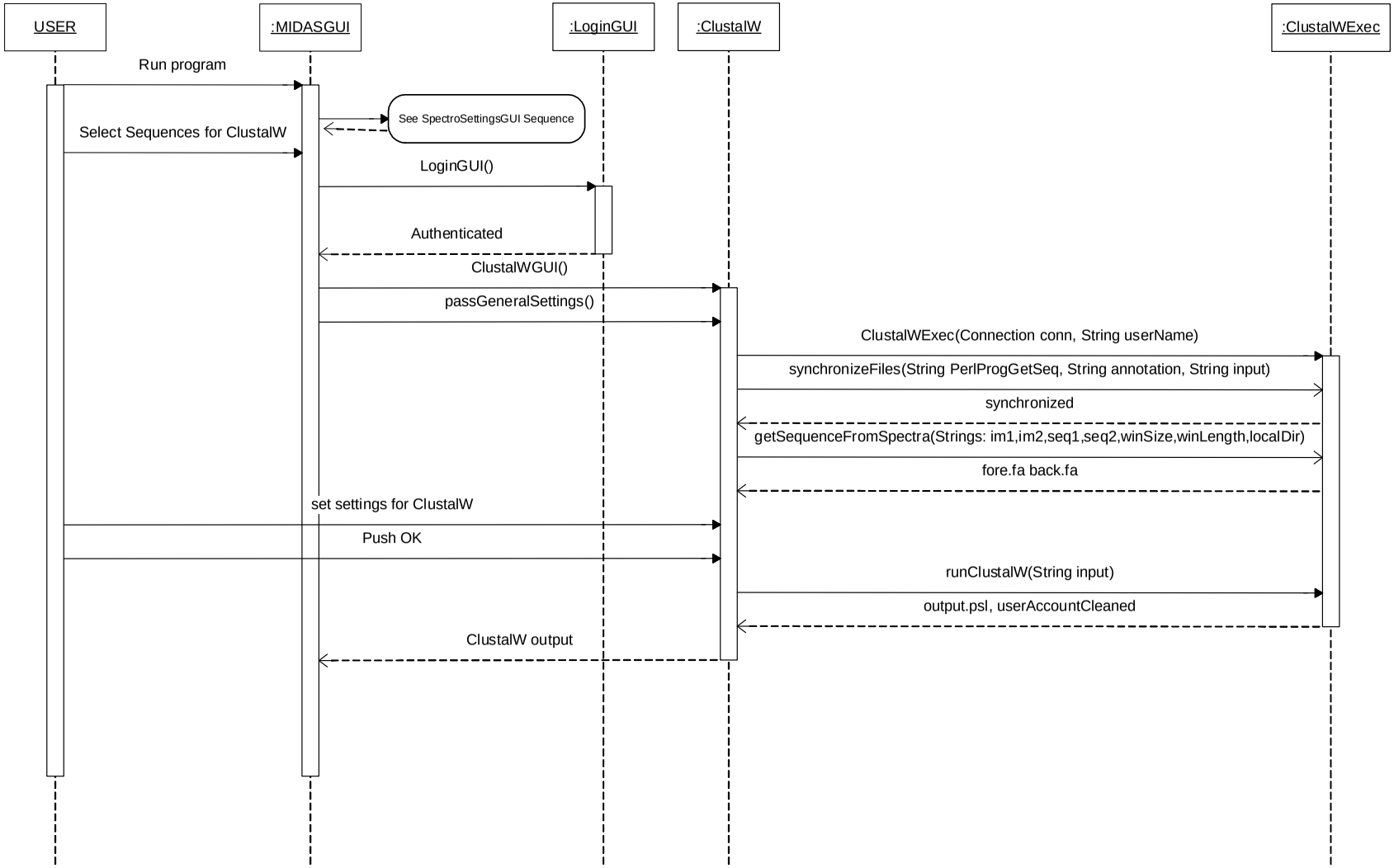


Figure 22 ClustalW Sequence Diagram

5.4 Interface Design

One challenge of this project is to visualize a lot of biological knowledge and data on one screen for analysis of DNA. This Section will give an impression of the layouts options discussed during this project for the visualization of the combination of Spectrogram Analysis and sequence alignment results.

At first one of the user's whishes was to have the output of a Spectrogram Analysis next to an output of a sequence alignment. The layout for the main output in MIDAS is illustrated in Figure 24.

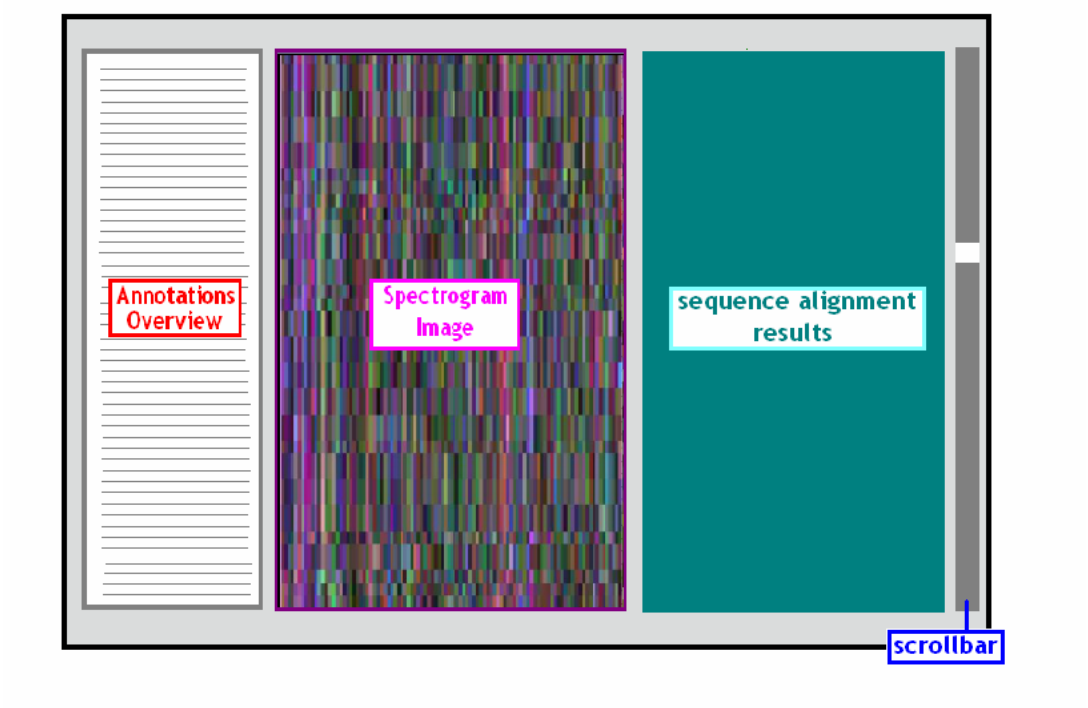


Figure 24 Storyboard 1

The annotations overview contains all the annotations corresponding on that current spectrogram image. The scrollbar can be used to scroll through the whole Spectrogram Analysis output. The sequence alignment results are displayed next to the spectrogram image. The disadvantage is combining this all on one screen makes the output panel very busy.

After a meeting where the requirements and design were discussed, it became clear that it is not a priority to have the sequence alignment results and Spectrogram Analysis output on one screen. Instead a visualization of the patterns on a spectrogram image has a higher priority. The visualization of the patterns can be done by visualizing a hierarchical clustered tree. The issue was how to display the clustering, by colored based lines (each color one hierarchy) as in Figure 25, or by drawing a hierarchical tree as in Figure 26.

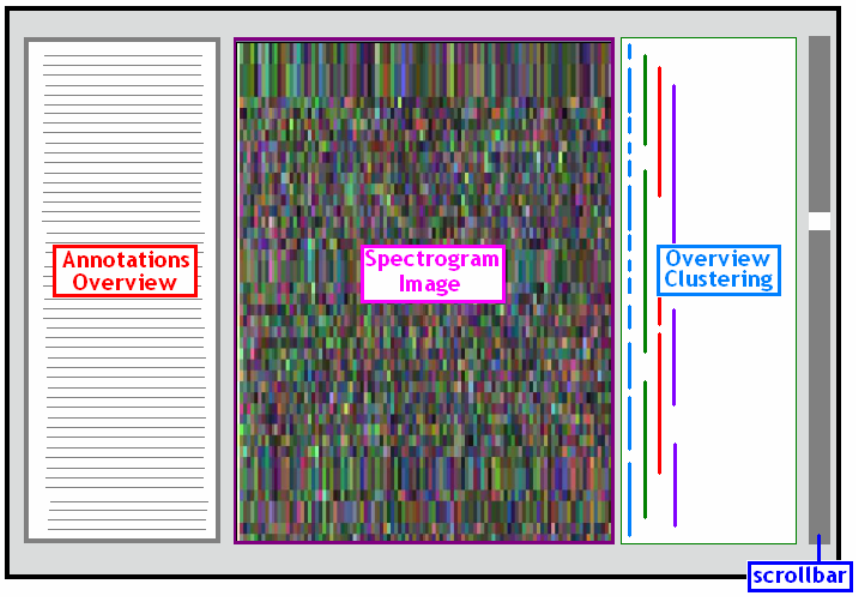


Figure 25 Storyboard 2

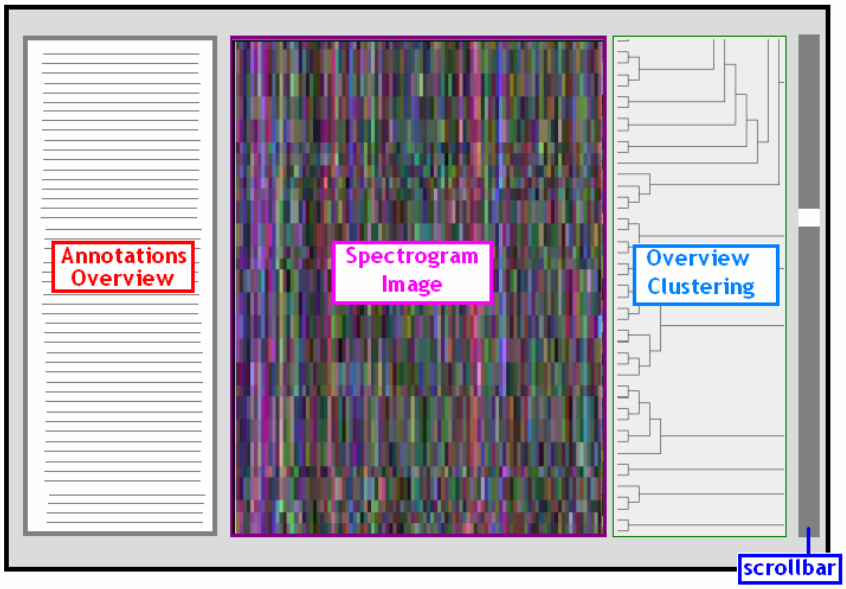


Figure 26 Storyboard 3

The final decision was to visualize the hierarchical clustered tree, as a hierarchical tree. In the hierarchical approach it is more clear to see which sequence belongs which clusters than in the color-based approach. Also the hierarchical tree is closer to reality of how a hierarchical clustered tree would look like.

The sequence alignment outputs are displayed on a separate output screen. Each new output screen, whether for Spectrogram Analysis or sequence alignment, is put on a new tab in MIDAS. The bioinformatics can analyze the biological data in one application, just by switching between tabs.

6 Implementation

In this chapter the implementation phase is discussed. Visualizing the hierarchical clustered tree is one of the innovations of MIDAS. The implementation and used algorithms are presented Section 6.1. In Section 6.2 the rewritten and new MATLAB functions can be found. The last Section 6.3 discusses the problems during this implementation phase and how they are solved.

6.1 Visualizing the hierarchical clustered tree

One of the most important requirements of this project is to visualize all the present clusters in a spectrogram for pattern analysis. In order to design and implement the visualization of the hierarchical clustering in MIDAS, it is needed to investigate how the hierarchical clustering is constructed in the Spectrogram Analysis in MATLAB. This is explained in the survey (6.1.1). As follows the implementation of the visualization of the hierarchical clustered tree is explained step-by-step(6.1.2).

6.1.1 Survey

In the Spectrogram Analysis the clustering is done binary and hierarchical. In this data structure each node always has two children, except the leaf nodes.

Constructing the binary hierarchical clustered tree is based on Euclidean distances. For computing the Euclidean distances the MATLAB function pdist(X, 'euclidean') is used.

Syntax

```
Y = pdist(X)
Y = pdist(X,distance)
Y = pdist(X,@distfun)
Y = pdist(X,'minkowski',p)
```

Description

`Y = pdist(X)` computes the Euclidean distance between pairs of objects in *n*-by-*p* data matrix `X`. Rows of `X` correspond to observations; columns correspond to variables. `Y` is a row vector of length $(n-1)\cdot n/2$, corresponding to the $(n-1)\cdot n/2$ pairs of observations in `X`. The distances are arranged in the order (1,2), (1,3), ..., (1,*n*), (2,3), ..., (2,*n*), ..., ..., (*n*-1,*n*). `Y` is commonly used as a dissimilarity matrix in clustering or multidimensional scaling.

To save space and computation time, `Y` is formatted as a vector. However, you can convert this vector into a square matrix using the `squareform` function so that element *i,j* in the matrix, where $i < j$, corresponds to the distance between objects *i* and *j* in the original data set.

`Y = pdist(X,distance)` computes the distance between objects in the data matrix, `X`, using the method specified by `distance`, where `distance` can be any of the following character strings that identify ways to compute the distance.

`'euclidean'`	Euclidean distance (default)

Figure 27 Syntax pdist MATLAB

The Euclidean distance is computed between two sequences of the Spectrogram Analysis. For N number of sequences there are (N*(N-1))/2 distances.

Based on these Euclidean distances a linkage table is constructed in MATLAB. The linkage table creates a hierarchical clustered tree.

Syntax

```
Z = linkage(Y)
Z = linkage(Y,'method')
```

Description

`Z = linkage(Y)` creates a hierarchical cluster tree, using the Single Linkage algorithm. The input `Y` is a distance vector of length $((m-1)\cdot m/2)$ by-1, where *m* is the number of objects in the original data set. You can generate such a vector with the `pdist` function. `Y` can also be a more general dissimilarity matrix conforming to the output format of `pdist`.

Figure 28 Syntax linkage MATLAB

In the linkage table we can find (N-1) linked sequences or group of sequences. To understand this, the basic definition of a linkage should be clear.

For example (Table 1):
There are 6 objects

7=	1	2
8=	3	7
9=	4	5
10=	6	8

Table 1 example linkage

The first 6 objects are paired up based on the Euclidean distance. Two sequences which have the lowest Euclidean distance are considered as a pair, thus are more similar. The pairs are new groups of objects, thus i.e. object 1 and 2 combined make group 7. As follows it is possible to have group 7 paired up with object 3. The linkage function continues to pair up objects with objects, objects with groups of objects or groups of objects with groups of objects based on the Euclidean distances till (N – 1) pairs exist.

In MIDAS' case the objects are sequences, the groups are clusters. Thus in this table each node and child can be found and to which cluster they belong.

As follows the permutation table is created in the Spectrogram Analysis. In the permutation table the new clustered order of the sequence indexes in the spectrogram can be found. This table gives an overview of what the original sequence index is of a new clustered sequence index.

With this information the visualization of a hierarchical clustered tree can be designed and implemented.

6.1.2 Hierarchical clustered tree visualization step-by-step

To retrieve the structure of the hierarchical clustered tree, it is needed to perform the next steps in order to construct a tree. This tree is constructed by recursively retrieving the cluster information from linkage table backwards. The algorithm for this search is as follows:

1. Look up next sequence from Spectrogram Image, which is found in permutation table, and remember the original sequence index.

2. Find all children (recursively) of this original sequence index the linkage table and append the cluster information in format file.
    i. Pre-order depth-first search algorithm

3. If there are more sequences, go to step 1.

4. Write the format file.

These steps are explained in more detail next.

1. Look up next sequence from Spectrogram Image, which is found in permutation table, and remember the original sequence index.

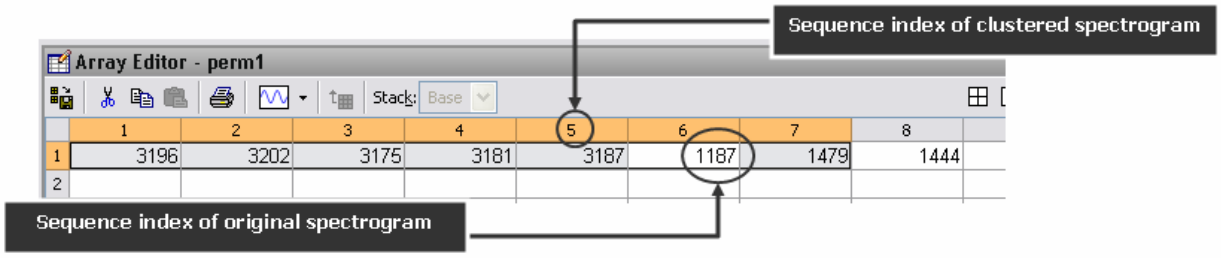


Figure 29 Permutation table

Figure 29 is an example of a permutation table. The column indexes correspond to the sequence index of the clustered spectrogram. The first row contains the original sequence index for each column, thus each clustered sequence index. The sequence index of the original sequence is used in the linkage table to construct the binary hierarchical tree. This leads to step 2.

2. Find all children of this original sequence index the linkage table and append the cluster information in format file.
    i. Pre-order depth-first search algorithm

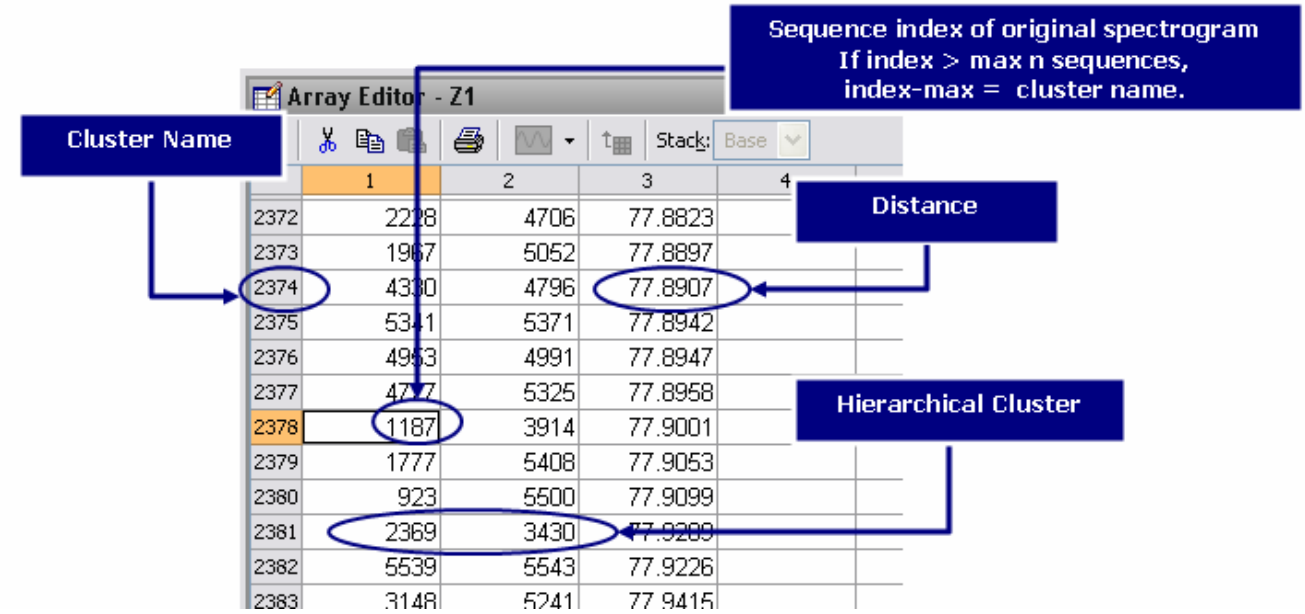


Figure 30 Linkage table

In the linkage table the paired up sequences, clusters or sequence with cluster can be found (see survey of this chapter for definition of linkage table). Clusters are represented by their cluster name. See Figure 30.

The original sequence index is found in the linkage table, and also the corresponding node C, which combines these two as a cluster. This corresponding node C can be another sequence, which makes this cluster a leaf. If the corresponding node C is another cluster, thus not a leaf and the tree has more branches. In this case the corresponding node C is bigger than the N number of Sequences. The branch can be found by (C-N), which is the cluster name of the

next branch. These steps are repeated till the leafs of one tree are found. The search algorithm used to construct the tree in this linkage table, is the pre-order depth-first search algorithm.

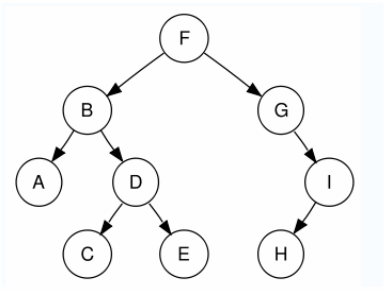


Figure 31 Pre-order depth-first

The search order of the pre-order depth first search algorithm is demonstrated in Figure 31. The order is (F,B,A,D,C,E,G,I,H).

The cluster names are appended in a format file. Each line number in the format file corresponds to the sequence index. Thus the first line in the format file is sequence 1, second line is sequence 2, etc. Each sequence belongs to one or several clusters. In the format file the cluster names belonging to one sequence are appended in the corresponding line.

3. If there are more sequences, go to step 1.

   For every sequence the corresponding clusters should be found.

4. Write the format file.

   The format file is written for the draw function in Java.

To illustrate this algorithm, see the next example:
For this example the smallRNAs.fa, first image, first seven sequences are used. smallRNA contains 3274 (=N) sequences .
The Spectrogram image is presented in Figure 32.

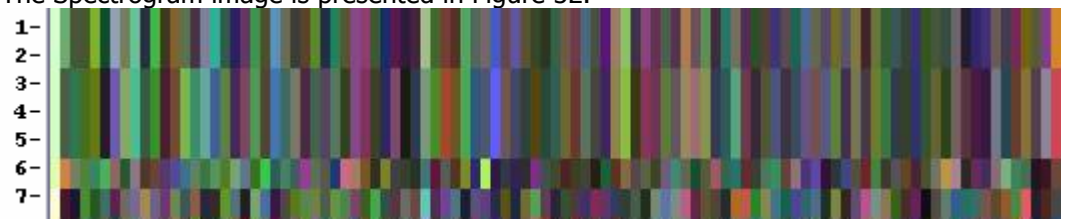


Figure 32 Partial Spectrogram Image

For sequence 1 the original sequence index is 3196 (Figure 29). By using the search method in the linkage table for the first sequence the binary tree in Figure 33 is constructed. Thus sequence 3196 is paired up with 3202 in the linkage table with cluster name 1 (row 1). The cluster name found is appended in the first line of the format file. Thus the first line contains the cluster name 1.

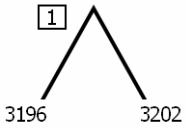


Figure 33 Tree sequence 1-2

For sequence 2 the original sequence index is 3202 (Figure 29). This cluster was already found in sequence 1, thus sequence 1 and sequence 2 are a cluster named 1. Thus the second line in the format file also contains the cluster name 1.

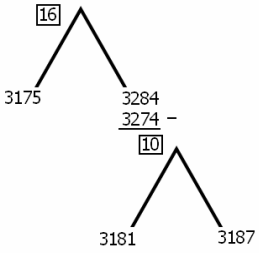


Figure 34 Tree sequence 3-5

Sequence 3 has the original index is 3175 (Figure 29). Using the search method the corresponding node is found: 3284, and 3284 > N (=3274). This means this sequence is paired up with another cluster, 3284-3274 = 10. The name of the cluster of which this sequence is paired up is 10. The next step is to go to row number 10 (rows are cluster names), then the nodes of this cluster are found. Cluster 10 is a leaf. The information appended in the format file is 16 and 10. Thus the

search ends here for sequence 3, and the following tree is found, Figure 334.

Step 1 and 2 are also performed for sequence 4, 5, 6 and 7. Sequence 4 and 5 make cluster 10 (part of Figure 33). For sequence 6 and 7 the trees in Figure 35, are constructed.

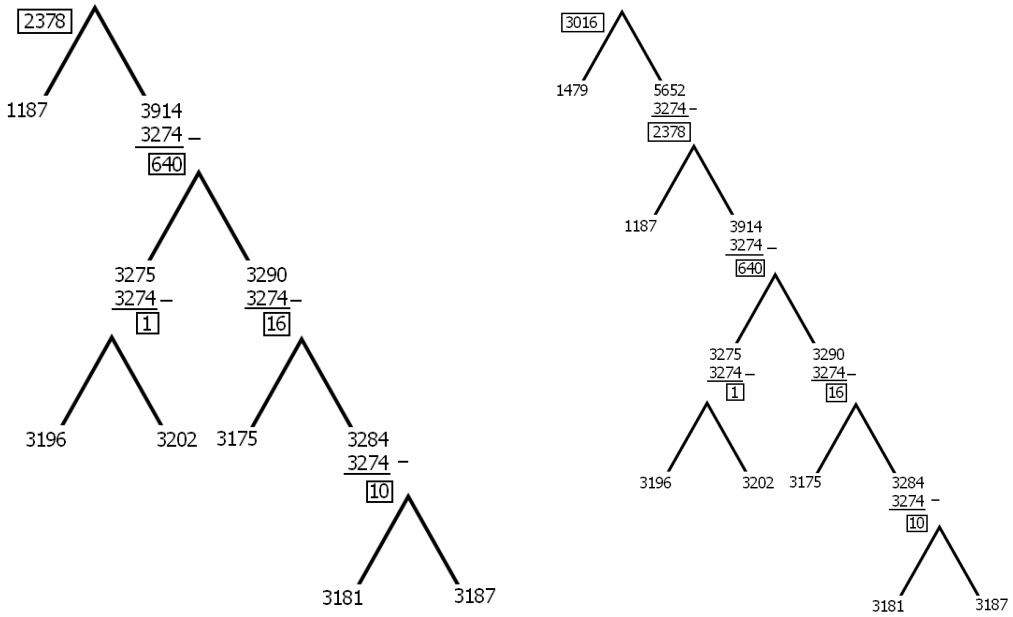


Figure 35 Tree sequence 6 and tree sequence 7

Java draw method

After these steps the format file is written, the result is visible in Figure 36, A. Here you can see the Spectrogram Image next to format file.

As said above, each line contains the clusters for that sequence. As you might notice, sometimes duplicates of a cluster name appear in the format file. This is done because that is how the Java draw method knows on which hierarchy level some cluster needs to be drawn. At first all the coordinates of the clusters are computed, and as follows the tree will be drawn. This approach has been chosen, because MIDAS displays 50 sequences at most at one image. For each image, MIDAS loads the corresponding cluster coordinates and draws the clustering for that image.

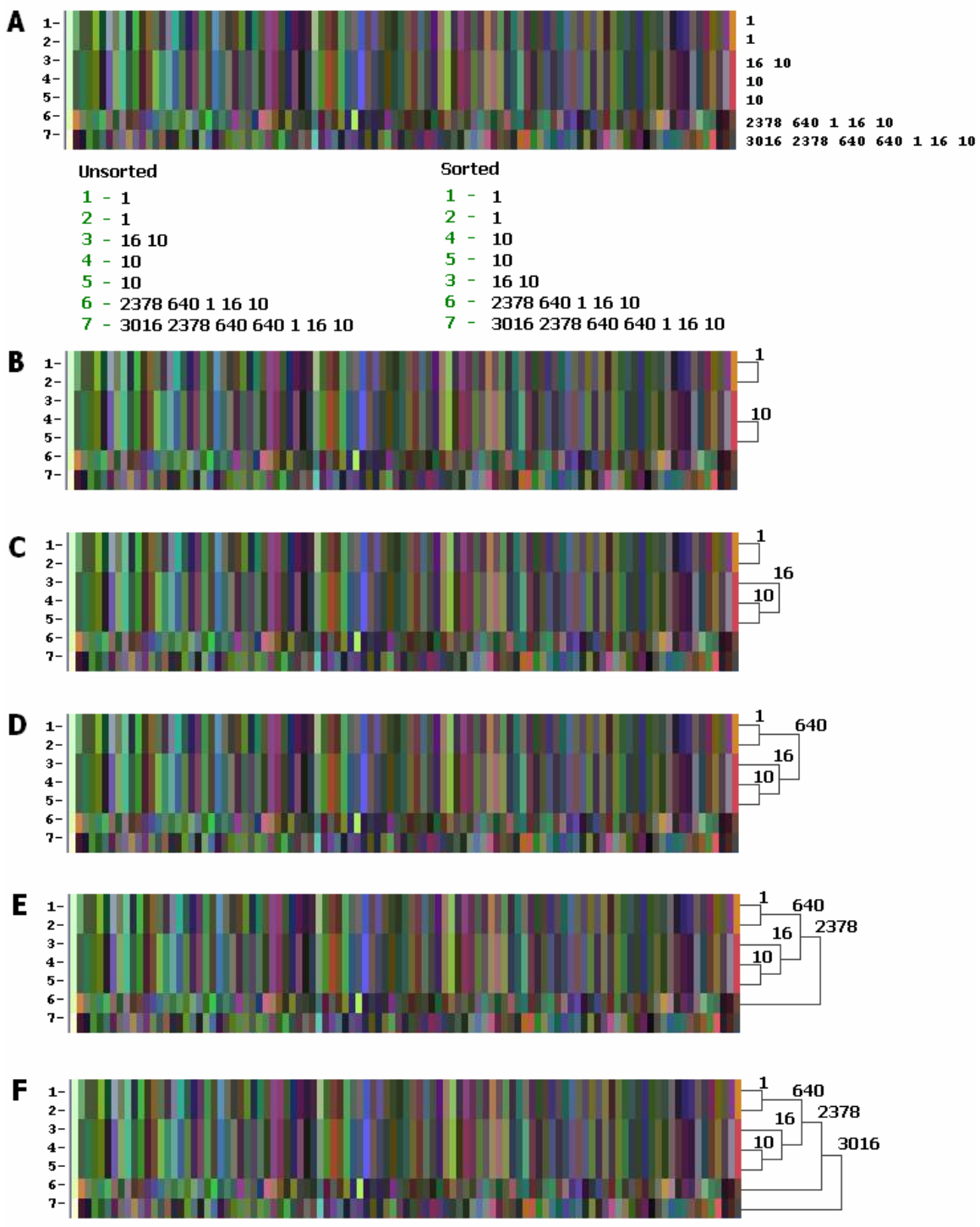


Figure 36 Java draw method

Figure 36 shows the unsorted format file. For the java draw method we first sort the file on number of clusters of each line, which results in the sorted format.

Next all the coordinates are computed. Because the file is sorted, first all the clusters which consist of a sequence-sequence relation are found. This is the lowest hierarchy of a hierarchical clustered tree. Next all the clusters with sequence-cluster relations are found. And

last but not least all clusters with cluster-cluster relations are found. By sorting the file, the hierarchy levels are known and all the coordinates can be computed correctly.

In order to draw the binary hierarchical tree all the information of a format file is put into hashmaps. Java hashmaps are a fast way to retrieve information without having to loop through any lists, which is beneficial performance wise. Hashmaps created are:

```
/*Hashmaps which are used to store the information we need to draw the clusters
 * clusters     : Hashmap containing the cluster names, with corrsponding x1,y1,x2,y2 coordinates
 *                These coordinates of one cluster are as follows a(x1,y1) b(x2,y2)
 *                  a---
 *                      |-----
 *                  b---
 *
 * lines        : Hashmap containing String Line number and String Cluster name.
 *                This hashmap only contains line numbers and corrsponding cluster names
 *                of the first hierarchy (cluster consisting of line-line combination)
 *
 * linesWithClusters    : Hashmap containing String line number and String[] corresponding clusters names
 *                        This hashmap contains the rest of the hierarchy levels.
 *
 */
```

Figure 37 Java comments on code

The clusters hashmap contains information build from the lines and linesWithClusters Hashmaps.

For computing the coordinates of the first hierarchy level the method searches for two sequences with the same cluster name. The information can be found in the lines hashmaps. With this the coordinates can be calculated (sequence to sequence on first level). The cluster name and coordinates are put in the clusters hashmap.

Computing the coordinates of higher hierarchy levels is more complex. A sequence which belongs to several clusters has several cluster names on his line.

This information is in the linesWithCluster hashmap. The line is read from right to left. Most of the cases, the clusters coordinates from the right are already computed. This is because of the bottom up approach; the clusters with sequence-sequence relation and sequence-clusters will occur earlier due to the fact that the format file is sorted. In case of a cluster with cluster-cluster relation, it is expected that the two following cluster names in the line are the nodes belonging to the cluster name, with the duplicates filtered. This is only NOT the case for the first cluster name (from left), because this is the cluster of the sequence paired up with the second cluster name on the line.

Duplicates are inserted because in some cases it is needed to repeat who the parent is of two nodes, due to the way the line is read. As soon as the cluster coordinates are computed, they are added to the clusters hashmap. If a cluster's coordinates are already computed, the next value is read.

To explain how a line is read with all the possible considerations, see the next example:
For example:

A 1356
1356
B 1323
1323
C 1274
1274
D 1039
1039
E 2247 2185 1640 1849 1640 1039 1274 1849 1323 1356

If the reading of line E starts from the right to left, A, B, C and D, coordinates are already computed.

With 1356 and 1323 as leafs and already computed, 1849 is the first "strange" cluster. In this case the two following are 1323 and 1356. Thus 1849 is cluster of 1323 and 1356.
Next 1274 and 1039 are also known clusters, thus we find 1640. 1640 consists of 1039 and 1039. As follows we find 1849 again. This value was added to the clusters hashmap as soon as the coordinates were computed, thus the next value is read. 1640 is also already known, thus next value.
2185 is a "strange" cluster, thus 2185 consists of 1640 and 1849. Next value is read.
2247 is the first value of the line. In this case cluster 2247 does NOT consist of 2185 and 1640. Because the first cluster name (from left) is the cluster of the sequence paired up with the second cluster name on the line. Thus 2247 is the cluster of sequence E and cluster 2185.

The format file in Figure 36 is a much simpler example. First the coordinates of cluster 1 and 10 are computed, and as follows 16, 640, 2378 and 3016. All the coordinates are now in the clusters hashmap. If the tree for line 1-7 is drawn in Java, first all the cluster names are collected of those lines. Next the tree is drawn hierarchically.
Figure 36 shows in step B that the first hierarchy level is drawn first. Which are lines 1-2, 4-5 in the format file. In step C the second hierarchy is drawn, line 3. In step D the third hierarchy is drawn. And in step E, F the fourth and fifth hierarchy, respectively, are drawn.

6.2 MATLAB functions

The preliminary work written by Evan Santo and Nevenka Dimitrova was able to create a DNA Spectrogram with the annotation of sequences. This work consisted of several MATLAB script files. In project MIDAS some of these files are reused and some new MATLAB functions are programmed. In Section 6.2.1 the MATLAB functionality is described for running a discontiguous or chromosome Spectrogram Analysis. Section 6.2.2 explains the file for creating a spectrogram video. Another file for creating partial spectrogram images is described in Section 6.2.3. Section 6.2.4 describes the MATLAB functions used by building a hierarchical cluster tree.

To get an understanding of the complete preliminary work we refer to the readme file of Spectrogram Analysis by Evan Santo.

6.2.1 Running a Spectrogram Analysis

To run a Spectrogram Analysis some changes are made to the preliminary work and some new files were created.

Changes:

ClusterChromosomeSpectro.m
Description

Preliminary summary:	This program reads the DNA sequence from a FASTA file and generates a clustered RGB image of the spectrogram output. This program calls createChromsomeClusteredSpectrogram.m. These two programs deal exclusively with large, contiguous sequences (i.e. whole chromosomes). The resulting image is saved to the hard disk, so it can later be processed by createClusteredSpectroVideo.m. This processing will turn the large image into an annotated movie according to the parameters in the createClusteredSpectroVideo.m file. Author: Evan Santo 6/5/2006 BMI
Changes:	-This code is adjusted so that it can be called from within JAVA. -CreateSpectroImages.m is called in this function to create the partial images for the scrollable spectrogram in MIDAS -This m-file is rewritten as a function so that it can be called from within Java

-The function call to createChromosomeClusteredSpectrogram.m is rewritten so that it can take arguments and give back also the freqDims. FreqDims is needed as an argument in CreateSpectroImages.m and the function call to createChromosomeClusteredSpectrogram needed the outputPath as an argument.
-Every output is placed in the outputFolder which MIDAS specifies.
-Some new arguments are added. For a complete view of the arguments a list is placed below.

Note: For more information about this code, check the original source code of Evan Santo. In these source files the full comments will explain his work.

Arguments

seqFile:	Path of input file (FA-format)
outputPath:	Path of folder where the output will be placed
stftWindowSize:	Length of a sequence that will be considered a window
stftWindowOverlap:	Overlap between adjacent windows
annotFile:	File were the annotation is placed
annotState:	Flag for annotation on(1) or off(0)
cluFlag:	Flag for clustering on(1) or off(0)
normalMethod:	Method for normalizing
numOfMean:	Numbers of mean
numOfStd:	Numbers of Std
annotClass:	Classes of features to annotate
annotType:	Tyes of features to annotate
opDelim:	Delimiting operator which is operating system specific
sequenceName:	The name of the input file
startPos:	This is the chromosomal coordinate that the sequence from the input file starts at
ori:	Orientation, can be either + or -
imageScale:	Scalar for scaling the image in createSpectroImages.m
numWin:	number of windows on one image in each frame in MIDAS
windowHeigth:	scalar to stretch the spectrogram vertically

CreateChromosomeClusteredSpectrogram.m

Description

Preliminary summary: This function returns a matrix of the spectrogram for the DNA sequence. This program is designed to operate exclusively for large, contiguous sequences in conjunction with ClusterChromosomeSpectrogram.m.

winSize	- width of the Short Time Fourier Transform (STFT) window
overlap	- the overlap between two consecutive STFT windows.
normalMethod	- method used for the normalization of the pixel values of the color spectrogram.
numOfMean,	- parameters for normalization.

numOfStd

The matrix is first clustered in STFT space. This clustering is then used to reference the RGB image so that the rows (windows) can be arranged in the correct order.

Changes: The outputpath is given as an input argument to this file and describes where the output must be placed.

NormalizeImage.m

Description

Preliminary summary: This function normalizes the RGB_Image such that all elements have values smaller than one. Two methods are supported: 'statisticalMax' or 'absoluteMax'.

Changes: Numerical values are casted to doubles because Java treats uncasted variables from MATLAB as integers.

CreateSpectroImages.m (New File)

Description

Summary: This function is called by ClusterDiscontiguousSpectrogram or by ClusterChromosomeSpectrogram in the last command and takes the RGB_Image. This RGB_Image is splitted into images of n windows each, and than saved in the output folder which the user specifies. These separate images are used by MIDAS to make the complete movie 'scrollable' in the interface.

Arguments

outputPath: to set the path where the images are placed

RGB_Image: to get the images from freqDims used in loop to know how much images are needed

imageScale: set the scaling of the images

numWin: Number of windows per screenshot

windowHeight: This sets the height of one window so that the annotation fits next to the window. The image is vertically stretch with this scalar.

ClusterDiscontiguousSpectro.m

Description

Preliminary summary: This program reads multiple DNA sequences from a FASTA file and generates a clustered RGB image of the spectrogram output. This program calls CreateDiscontiguousClusteredSpectrogram.m. These two programs deal exclusively with multiple sequences provided in FASTA format with headers that are formatted as described below. This will be primarily of interest when wanting to obtain higher resolution and more focused studies of specific features annotated in genomes. Allowing for cross-species and interchromosomal comparisons to be made. The resulting image is saved to the hard disk, so it can later be processed by createClusteredSpectroVideo.m This processing will turn the large image into an annotated movie according to the parameters in the createClusteredSpectroVideo.m file.
Author: Evan Santo 6/26/2006

FASTA-format: >Species_Prefix:chr#:'+' or '-':StartPos-EndPos
sequence (atcgtct...)

>Next sequence header...

Changes: -This code is adjusted so that it can be called from within JAVA.
-CreateSpectroImages.m is called in this function to create the partial images for the scrollable spectrogram in MIDAS
-This m-file is rewritten as a function so that it can be called from within Java
-The function call to createDiscontiguousClusteredSpectrogram.m is rewritten so that it can take arguments and give back also the freqDims. FreqDims is needed as an argument in CreateSpectroImages.m and the function call to createDiscontiguousClusteredSpectrogram needed the outputPath as an argument.
-Every output is placed in the outputFolder which MIDAS specifies.
-This m-file now always creates a text file of the short annotation, because this is used to get the sequence out of a spectrogram by using the Perl program getSequenceFromSpectra.pl in MIDAS. This programs this this annotation file as input. If a user selects annotationstate ON, than the full annotation text is also generated in AnnotationFULL.txt
-Some new arguments are added. For a complete view of the arguments a list is placed below.

Note: For more information about this code, check the original source code of Evan Santo. In these source files the full comments will explain his work.

Arguments

seqFile:	Path of input file (FA-format)
outputPath:	Path of folder where the output will be placed
stftWindowSize:	Length of a sequence that will be considered a window
stftWindowOverlap:	Overlap between adjacent windows
annotDir:	Directory were the annotation files are placed
annotState:	Flag for annotation on(1) or off(0)
cluFlag:	Flag for clustering on(1) or off(0)
normalMethod:	Method for normalizing
numOfMean:	Numbers of mean
numOfStd:	Numbers of Std
annotClass:	Classes of features to annotate
annotType:	Tyes of features to annotate
opDelim:	Delimiting operator which is operating system specific
sequenceName:	The name of the input file
imageScale:	Scalar for scaling the image in createSpectroImages.m
numWin:	number of windows on one image in each frame in MIDAS
windowHeigth:	scalar to stretch the spectrogram vertically

CreateDiscontiguousClusteredSpectrogram.m

Description

Preliminary summary: This function returns a matrix of the spectrogram for the DNA sequences given. This program is designed to operate for

multiple smaller sequences after being read in from a FASTA file formatted as specified in ClusterDiscontiguousSpectrogram.m. There are many differences between the CreateChromosomeClusteredSpectrogram.m and this function, many having to do with the way annotation is obtained for each sequence.

winSize	- width of the Short Time Fourier Transform (STFT) window
overlap	- the overlap between two consecutive STFT windows.
normalMethod	- method used for the normalization of the pixel values of the color spectrogram.
numOfMean, numOfStd	- parameters for normalization.

The matrix is first clustered in STFT space. This clustering is then used to reference the RGB image so that the rows (windows) can be arranged in the correct order.

Changes: -The outputPath is given as an input argument to this file and describes where the output must be placed.
-This m-file now always creates a text file of the short annotation, because this is used to get the sequence out of a spectrogram by using the Perl program getSequenceFromSpectra.pl in MIDAS. This programs this this annotation file as input. If a user selects annotationstate ON, than the full annotation text is also generated in AnnotationFULL.txt

CreateSpectroImages.m(New File)
This script-file is explained earlier in this Section.

NormalizeImage.m
This script-file is explained earlier in this Section.

6.2.2 Creating a spectrogram video

CreateClusteredSpectroVideo.m
Description

Preliminary summary: This routine will create a Spectrogram video from 1 large RGB image and a corresponding annotation track as created and formatted by the ClusterChromosomeSpectrogram or ClusterDiscontiguousSpectrogram.m programs.
Usually, the Linux machines with the massive memory are used to create the clustered spectrogram image and annotation track. These variables are then dumped to disk by a file writer. These saved files then have to be dynamically loaded into the workspace on a Windows machine, so the large image can be processed into multiple frames and rendered as one video. Author: Evan Santo

Changes: -This code is adjusted so that it can be called from within JAVA.

-This m-file is rewritten as a function so that it can be called from within Java.
-This function is able work with arguments which are needed for MIDAS

Arguments

imagePath:	Path of the Image.txt
annotPath:	Path of the Annotation.txt
outputPath:	Path of the output folder where the video will be placed
movFileName:	The name of the movie which is created

6.2.3 Running BLAT or ClustalW

The output of the alignment is visualized on a tabbed panel in MIDAS. This tab contains the output of the alignment in text format and the annotation next to the partial image of the cluster which was selected by the user as input for the alignment. This partial image is created with the new MATLAB function CreatePartialImage.m.

CreatePartialImage.m (New File)

Description

Summary:	This function is called from within JAVA to create a partial image. This image is displayed together with the output of ClustalW. This function creates a partial image indexed by startPos and endPos. These positions are given as paramaters from the JAVA interface. The user selects clusters in this interface. The first index of this cluster is startPos and the last index is endPos. This program has the restriction that there is a maximum of 2 images can be taken to create a single image from. This restriction is checked in JAVA MIDAS) before making calls to this function. The reason why you can give at max 2 images, is that if you take more than 2 the image becomes to big to fit the outputscreen in MIDAS. If you accidentally give more images as arguments (so endIm-startIm > 2) it just pastes the first image to the last image and you end up with 2 images into one large image.

Arguments:

inputPath:	The full path of the input image (including filename and extension
outputPath:	The full path of the output image (including filename and extension.
startIm:	first image to start with
endIm:	last image to end with (startIm and endIm can be the same if there is only 1 image.
startPos:	start position indexed in the large image
endPos:	end position indexed in the large image

6.2.4 Visualize the hierarchical clustered tree

To visualize to hierarchical clustered tree in MIDAS two MATLAB script files are developed to build a text file with a special format in order to pass the cluster information to the Java draw method which draws this tree.

BuildClusterInfo.m (New File)

Description

Symmary:	It takes each value of the permutations table and calls findClustersCorresponding2Perm in a loop. When this function has completed, the outputfile clusterNamesTree.txt is written and ready to be drawn in MIDAS in JAVA.

Arguments

linkagePath:	Path of the linkage table
permutationsPath:	Path of the permutations table
outputPath:	Path where the files should be outputted
maxRecursionCounter:	Maximum number of recursions (for more information please read the comments on that part in findClustersCorresponding2Perm.m

FindClustersCorresponding2Perm.m (New File)

Description

Symmary:	This function is called by buildClusterInfo.m and searches recursively the structure of the hierarchical tree and builds a string array which is passed to buildClusterInfo.m This string array contains the nodes and the leaves of one permutation value.
Function:	This function searches all corresponding clusters of one permValue. This linkage table contains three colums: two which contain the nodes or leaves of the tree, one with the distance which is not of interest for us. The permValue can be found in the linkage table in one row. In the other column of the same row the corresponding cluster can be found. This clusterValue refers to the other rowNumber of the linkage table which contains the other corresponding clusters. The search for the other corresponding clusters is done recursively until no other clusters can be found.
Algorithm:	For walking through the tree, we used a depth-first search algorithm. The leaves are searched in pre-order.

Arguments

permValue:	The value in the permutation table.
linkagePath:	This is the linkage Table created by MATLAB
permutationsPath:	This is the permutation Table created by

Output

Out	out contains a string with clusterNames corresponding to one permValue. A clusterName is a rowNumber in the linkageTable.

6.3 Problems and solutions

During this project several problems occurred. Problems like Java bugs, performance issues and inconsistencies in object casting. In this chapter the problems are presented and the solutions.

Java look and feel inconsistencies

MIDAS has a Windows look and feel, except for the browse dialogs. This is due to the inconsistencies with the jFileChooserDialogs in Java 1.5. There are a number of known jFileChooserDialogs bugs at Java. If the look and feel for the browse dialogs are set on the Windows look and feel the Java Virtual Machine crashes on a problematic frame.

The solution is to set the look and feel for the browse dialogs on Java look and feel when they are initialized, and set the look and feel back on Windows if the browse dialog is closed.

MATLAB object casting

MATLAB computes with variables without cast typing, the variables are treated as double(0.00) values. When the MATLAB scripts are converted to Java, these variables are treated as Integer(0) objects . In the Spectrogram Analysis scripts a lot of computation and checks are performed on these double variables. Only in the script files, the objects are not casted to double. Variable casting in MATLAB is needed for Java. Computations and checks might work in MATLAB, but not in Java if some are treated as Integers instead of doubles.

The solution is to cast every object in MATLAB for consistency in Java.

Java Hotspot Virtual Machine crashes

If a user runs a method on the MATLAB Component Runtime Machine, and he gives incorrect inputs concerning content, which will not cause a MATLAB error, the MATLAB function will not have a correct or no result. This will cause the Java Virtual Machine to crash and give a Java Hotspot error. This is because the Java code expects some output of MATLAB, only no correct result will be provided. Thus the Java Virtual Machine crashes.

This problem is actually also a constraint on MIDAS. The solution is to provide error handling on this level, but then expert biological knowledge is needed. MIDAS expects its users to be expert users.

Java heap space memory problems

The first approach of visualizing all the spectrogram images was to have all the images scrollable. Thus the user could scroll from image 1 to the last image. This approach demands all the images to be loaded at once. The Java heap space memory cannot take the size of the amount of large images to be loaded.

One solution is to expand the Java heap space memory. This solution is not the best solution for stand-alone application, thus another approach was implemented. The other approach is to use a Java slider. With this approach only one image will be loaded at one moment, thus the heap space memory will not be exceeded. Only disadvantage is that the slider does not "flow" through the images, but reloads each image.

7 Testing and Evaluation

This chapter describes the Testing and Evaluation phases. These are very important phases in software development. Testing and Evaluation allows the developers to analyze their application and improve if necessary.

In Section 7.1 the testing phase is described. In Section 7.2 the usability tests are described. And in Section 7.3 the constraints of MIDAS are found.

7.1 Testing

Software testing is the process used to measure the quality. Aspects as correctness, completeness, security, capability, reliability, efficiency, maintainability, compatibility, and usability in/of the system are important during the testing phase.

Testing in this project is done by *GUI testing* and *source code testing*. The source code is tested with own test functions, and also by the use of JUnit (regression testing framework). The following detailed test cases can be found in Appendix C:

- Input Chromosome for analysis
- Input Discontiguous DNA for analysis
- Use Spectrogram Analysis
- Create Spectrogram Video
- Watch Spectrogram Video
- Overview all present clustering DNA
- Login to Philips server
- Use BLAT
- Use ClustalW
- Load existing project
- Run BLAT without existing project
- Run ClustalW without existing project
- Disconnect from Philips Server

Source code testing is continuously used during the development of MIDAS. The major quality aspects of the integration of different tools in one application are reliability and correctness. For example, in MIDAS, part of the output of the Spectrogram Analysis is used as input for ClustalW. It is necessary that the part of the output of the Spectrogram Analysis is reliable and correct in order to perform multiple sequence alignment. In this example source code testing is used to provide reliability and correctness.

With GUI testing little inaccuracies are discovered. Examples are wrong warning message pops up when clicking a button and empty help files. GUI testing measures the usability, consistency and capability of the system.

7.2 Usability test

In the evaluation phase it is very important to evaluate your product using usability tests. If these participants are the actual end users of your product, they can give feedback so that the product fits to their needs. Since MIDAS is developed in an iterative incremental software process, each iteration needs an evaluation in order to see if the last iteration is reached or that iteration is needed to improve the product.

The usability test of MIDAS consists of two parts: Observed usability test (7.2.1) and the unobserved usability test (7.2.2).

7.2.1 Observed usability test

The participants during the last evaluation phase where the actual end users of MIDAS: the bioinformatics of Philips Research Bangalore.

A set of assignments was given the participants, including a manual on how to complete these assignments. After each assignment, they were asked to give their feedback. The participants were also observed during their test session in order to investigate their behaviour; the errors made and how intuitive MIDAS actually is. An overview of these results is given below.

The results from this usability test were very positive and a lot of ideas came up during these sessions to improve, correct or extend MIDAS.

Since it was not possible to implement each specific need of the user, choices have been made. To get a deeper understanding of all the needs and their priority a MoSCoW analysis can give the answer. In the Improvements, chapter 9, a MoSCoW analysis is given for the next iteration to MIDAS 2.0.

The test persons were asked to complete the following assignments:

Install MIDAS

Feedback:	Indeo5 codec pack should be included in the installer Java runtime environment should be included in the installer
Suggestions:	-

Run New Spectrogram Analysis

Feedback:	Use correct biological names in the panels Back button for changes in your settings
Suggestions:	Show the ranges of the settings on the panel Create a windows based menu

Output panel for Spectrogram Analysis

Feedback:	Separate the BLAT panel and the ClustalW panel from each other.
Suggestions:	Make the cluster tree clickable to select a cluster

Spectrogram Video

Feedback:	It works perfectly fine
Suggestions:	Make it possible that you do not need to wait during video creation but let it run in the background

Load Spectrogram Analysis Project

Feedback:	It works perfectly fine
Suggestions:	-

7.2.2 Unobserved usability test

Besides the observed usability test other participants worked with MIDAS unobserved on their own desktop. Different results were obtained from the unobserved test.

The feedback in this test was that MIDAS did not create the output in the way they were used to receive the output. There was a need for better filenames and folder names. From the name of the file and the folder, the user must be able to see what kind of DNA is used to create this output and what settings were set.

In general this test resulted positive and the user was able to give detailed feedback and new ideas. There were also some new bugs found in this test which were not found during earlier test sessions.

7.3 MIDAS Constraints

As MIDAS is in his first stage, there are still some constraints for the system. Most of these constraints can be solved by an improved solution. Some constraints require a workaround.

The constraints of MIDAS are summarized here below. A workaround or improved solution is described for each constraint.

- The MCR MATLAB Component Runtime runs on his own thread. The Java Virtual Machine is not able to interrupt this thread. Thus premature termination of MCR MATLAB Component Runtime is not possible.

  Workaround: Manually kill the process.

- Locating and fetching files in MIDAS is based on predetermined hard coded names. Inconsistency may occur when filenames or files in one directory that MIDAS uses are manipulated.

  Improved solution: Generate unique filenames in Java.

- Errors, Exceptions, JVM hotspot crashes during MCR execution, transfer between Server and local machine failures will be shown on the console. These errors are mostly caused by incorrect input arguments. On this level MIDAS do not provide error handling.

  Improved solution: To handle these errors expert knowledge is needed. Only the Bioinformatics know the correct input combinations.

- It is not possible to run BLAT or ClustalW on Chromosome Spectrogram Analyses. The code used for extracting Sequences of Spectrograms (getSequenceFromSpectra.pl by Evan Santo) only works for Discontiguous Spectrogram Analyses. The BLAT and ClustalW function for Chromosome are disabled.

  Improved solution: getSequenceFromSpectra.pl should be extended to work for Chromosome Spectrogram Analysis.

- The clustered hierarchical tree from Spectrogram Analysis can be very extensive, as some sequences have above 500 levels. The cluster tree in MIDAS is build recursively in MATLAB. The MCR MATLAB Component Runtime machine can only compute 500 recursions. Thus the cluster tree build in MIDAS is restricted to 200 recursions. A sequence with more than 200 levels will not be drawn.

  Improved solution: change the format of the input file, on which the tree is build.

- Loading an external Spectrogram Analysis is constrained to the format of a MIDAS Spectrogram Analysis output directory. This means that certain files and directories have to exist for MIDAS in order to provide the visualization.

  There is no improved solution or workaround for this constraint.

- Loading a project from a location with spaces in its path will cause problems when the Microsoft Wordpad is initialized in order to view a stacktrace or annotations.

  Workaround: Search the files manually and open them.

- Sliding through a Spectrogram Analysis, with full annotation is a heavy process.

  There is not yet an improved solution or workaround for this constraint.

8 MIDAS

MIDAS, Multimodal Interface for DNA Alignment of Sequences, is a stand-alone application which integrates Spectrogram Analysis and sequence alignment to analyze DNA, also providing a visualization of patterns in spectrograms.

In this chapter the features, along with screenshots, of MIDAS are introduced. Examples of features are run a Spectrogram Analysis and perform sequence alignment, load external spectrogram projects. For detailed information about how to use MIDAS, and its features, the MIDAS manual can be found in Appendix D.

As MIDAS is a stand-alone application, it is provided with an installer. MIDAS requires a Windows platform and a 1280x1024 screen resolution.

The start screen of MIDAS is presented in Figure 38. All the features can be found in the menu bar. Spectrogram Analysis, sequence alignment, Connection to and form server and view help files.

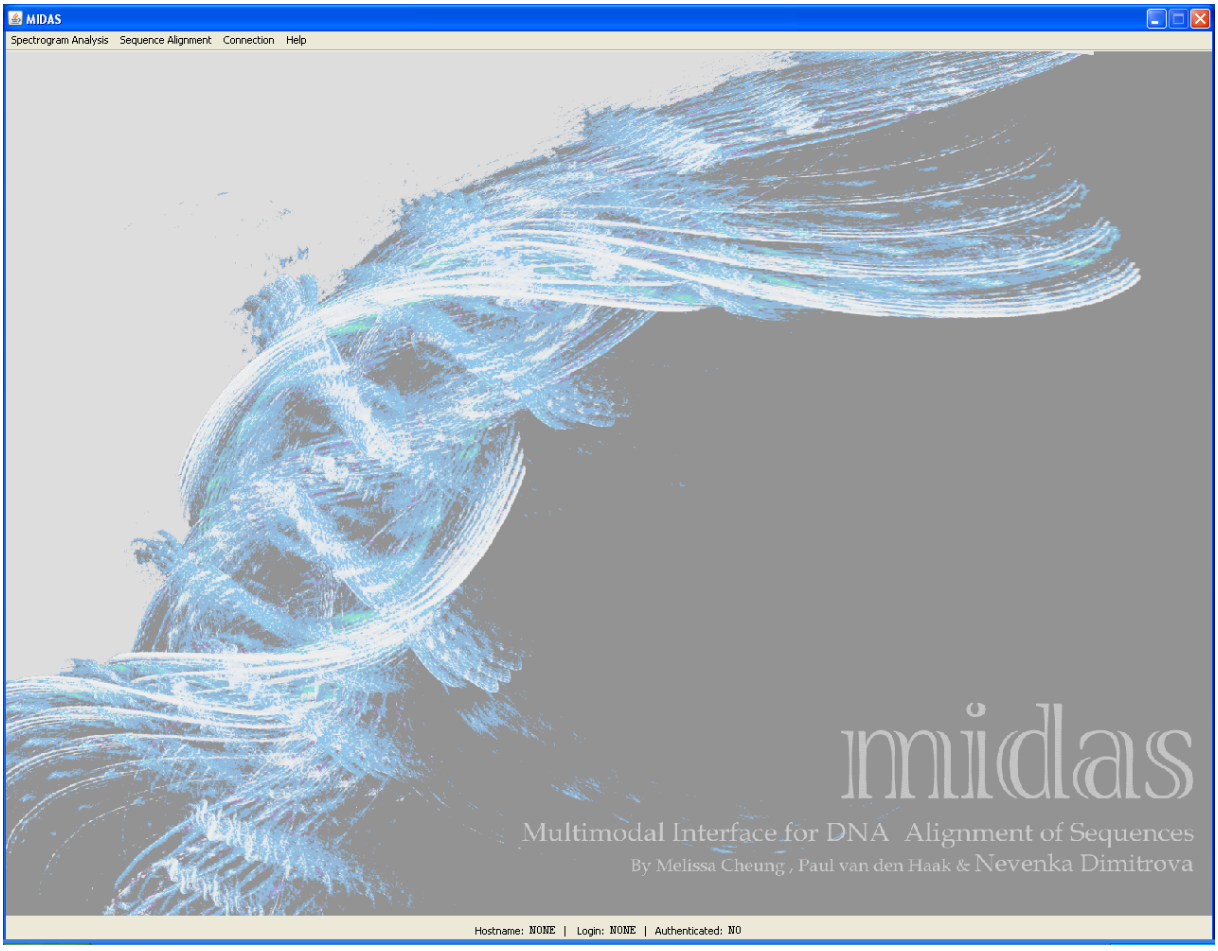


Figure 38 MIDAS start screen

Spectrogram Analysis can be started from the menu bar of the start screen. If this option is selected the Spectrogram Analysis settings will be loaded (Figure 39, Figure 40). It is possible to run a Discontiguous or Chromosome Spectrogram Analysis. In this settings panel all the input arguments can be given as in the original Spectrogram Analysis.

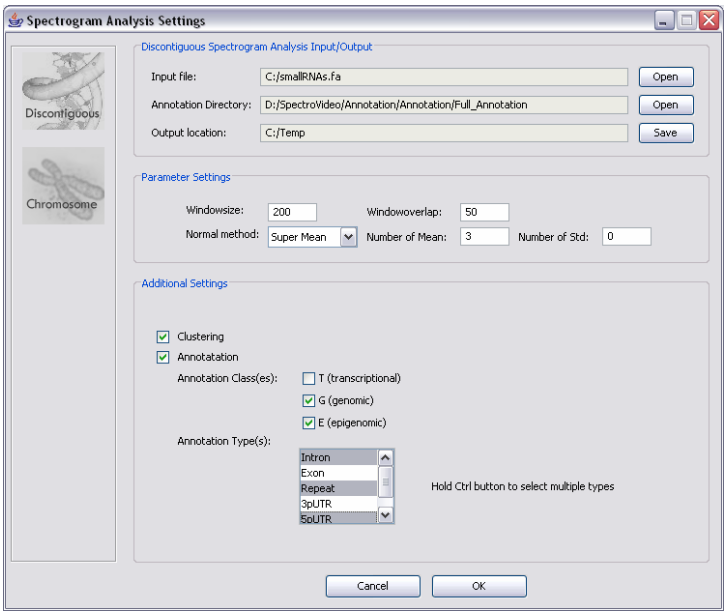


Figure 39 Discontiguous Spectrogram Analysis

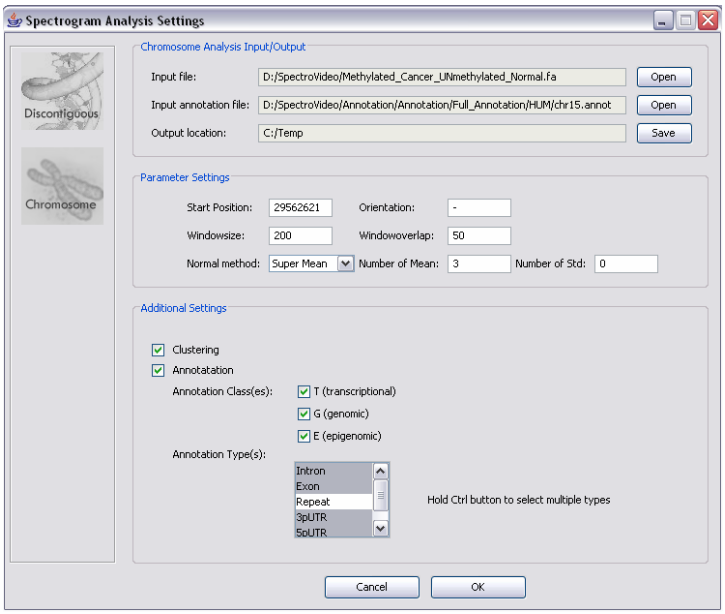


Figure 40 Chromosome Spectrogram Analysis

The output of a Chromosome Spectrogram Analysis is presented in Figure 41. The annotation overview with the annotations next to corresponding the spectrogram image, also if the spectrogram is clustered, the hierarchical clustered tree is drawn. The slider can be used to scroll through all the images of the Spectrogram Analysis. The output of a discontiguous Spectrogram Analysis is presented in Figure 42.

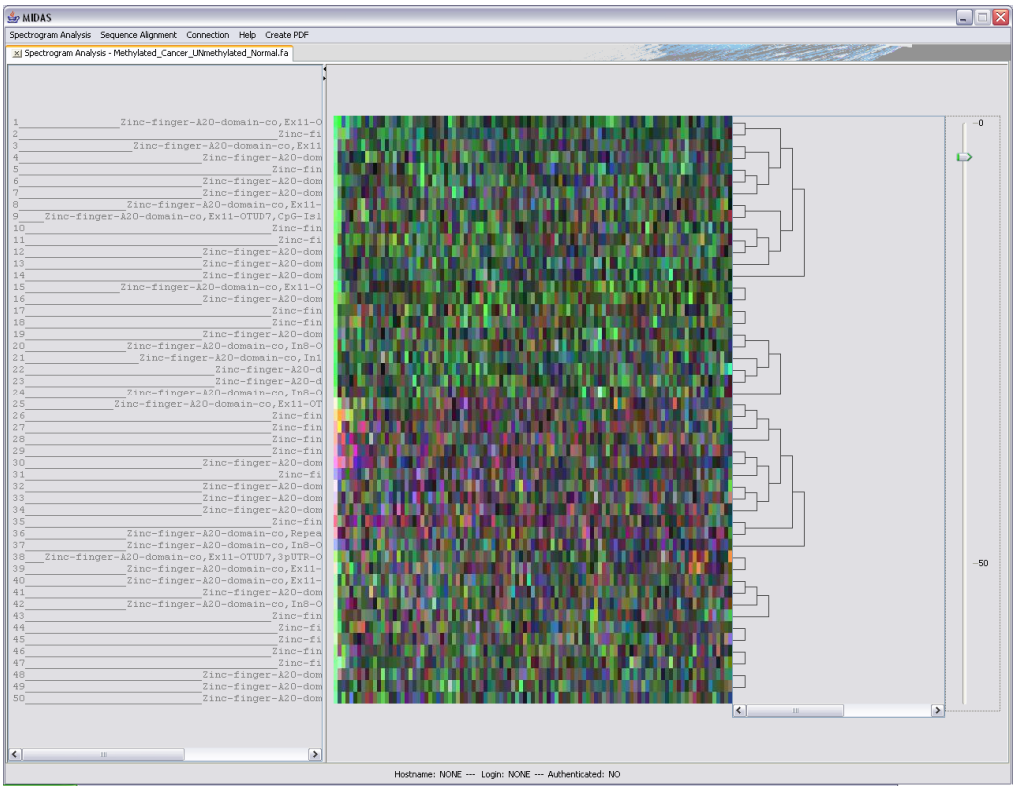


Figure 41 output chromosome spectrA

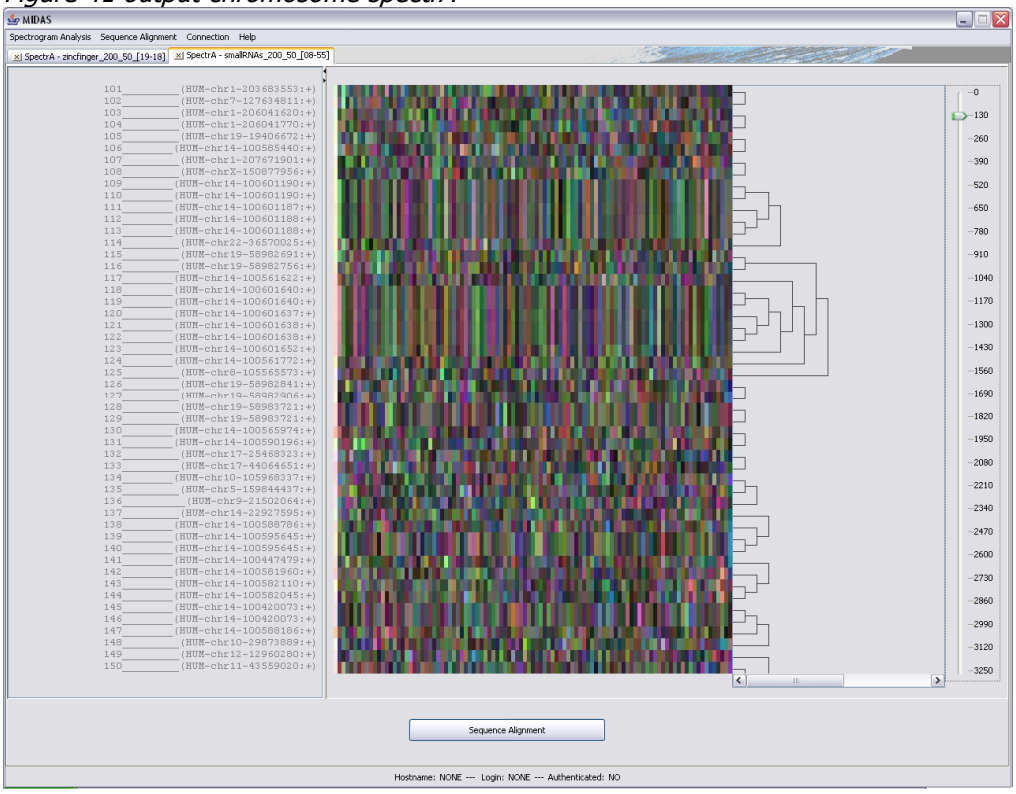


Figure 42 output discontiguous SpectrA

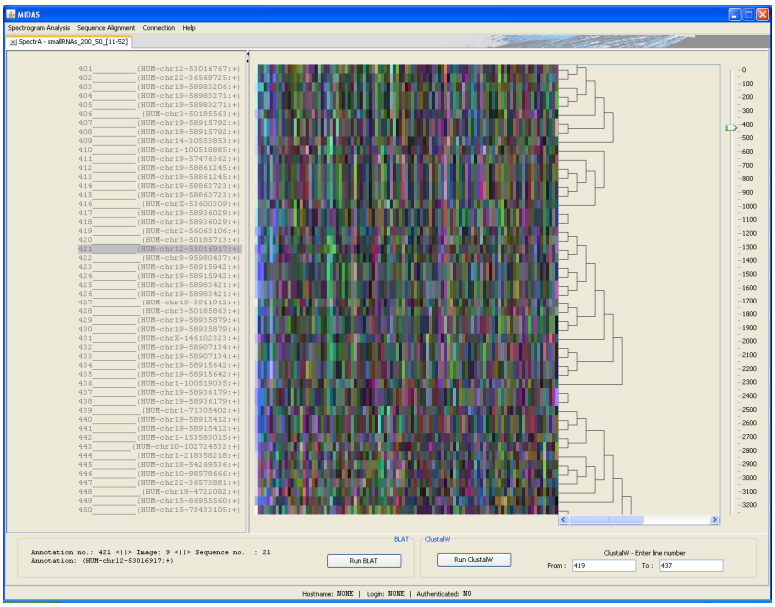


Figure 43 output discontiguous SpectrA

It is possible to perform sequence alignment on the output of the discontiguous Spectrogram Analysis (Figure 43). For sequence alignment in BLAT select one annotation in the annotation overview. The selection for BLAT is displayed in the lower left corner of the panel. For the selected annotation the BLAT settings panel is loaded (Figure 44). All the settings which can be set for BLAT can be set in this panel. For multiple sequence alignment an input range can be given to run ClustalW. For the selected input range the ClustalW panel is loaded (Figure 45). All the additional settings can be set in this panel. The BLAT and ClustalW tool are located on the Philips Server. If the user is not connected and authenticated yet, that has to be done first (Figure 46).

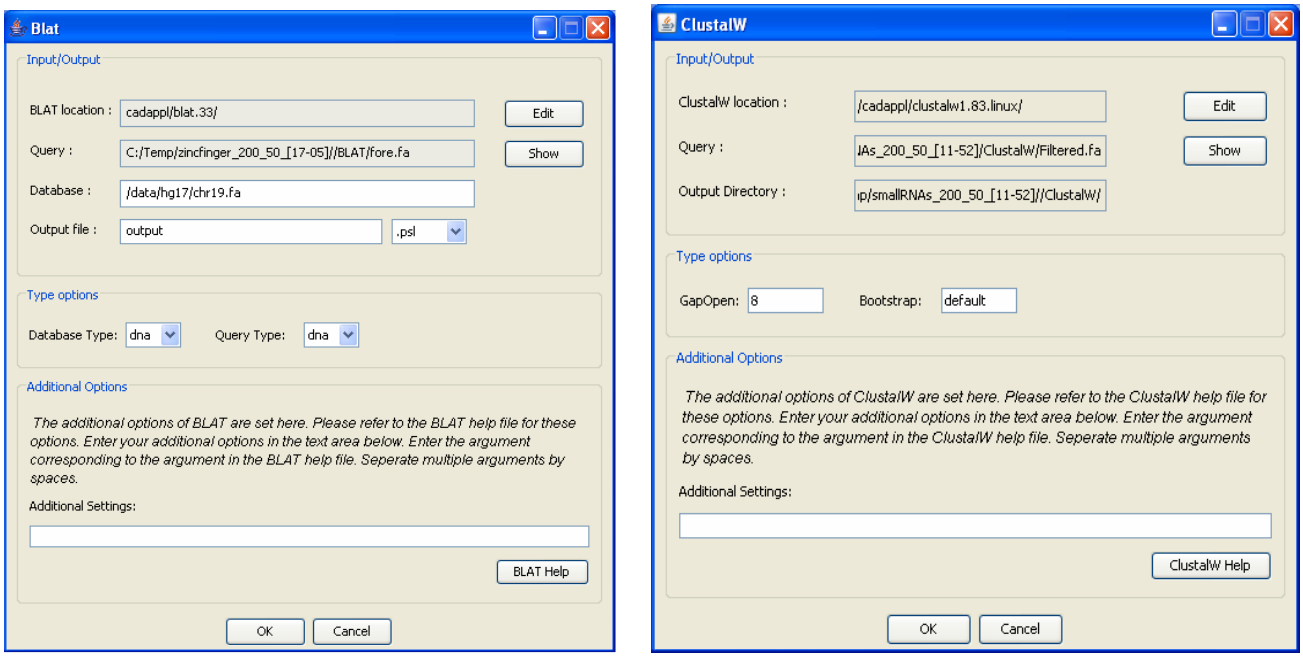


Figure 45 ClustalW settings panel

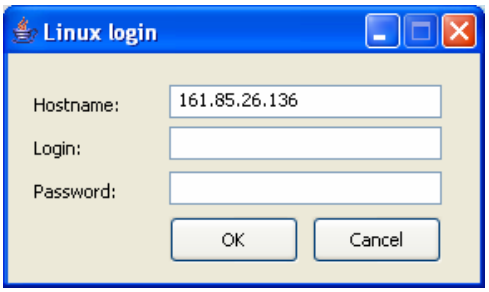


Figure 46 Linux login

The output of BLAT is presented in Figure 47. It is possible to write the output to a file, thus save the output on the desktop pc (Figure 48).

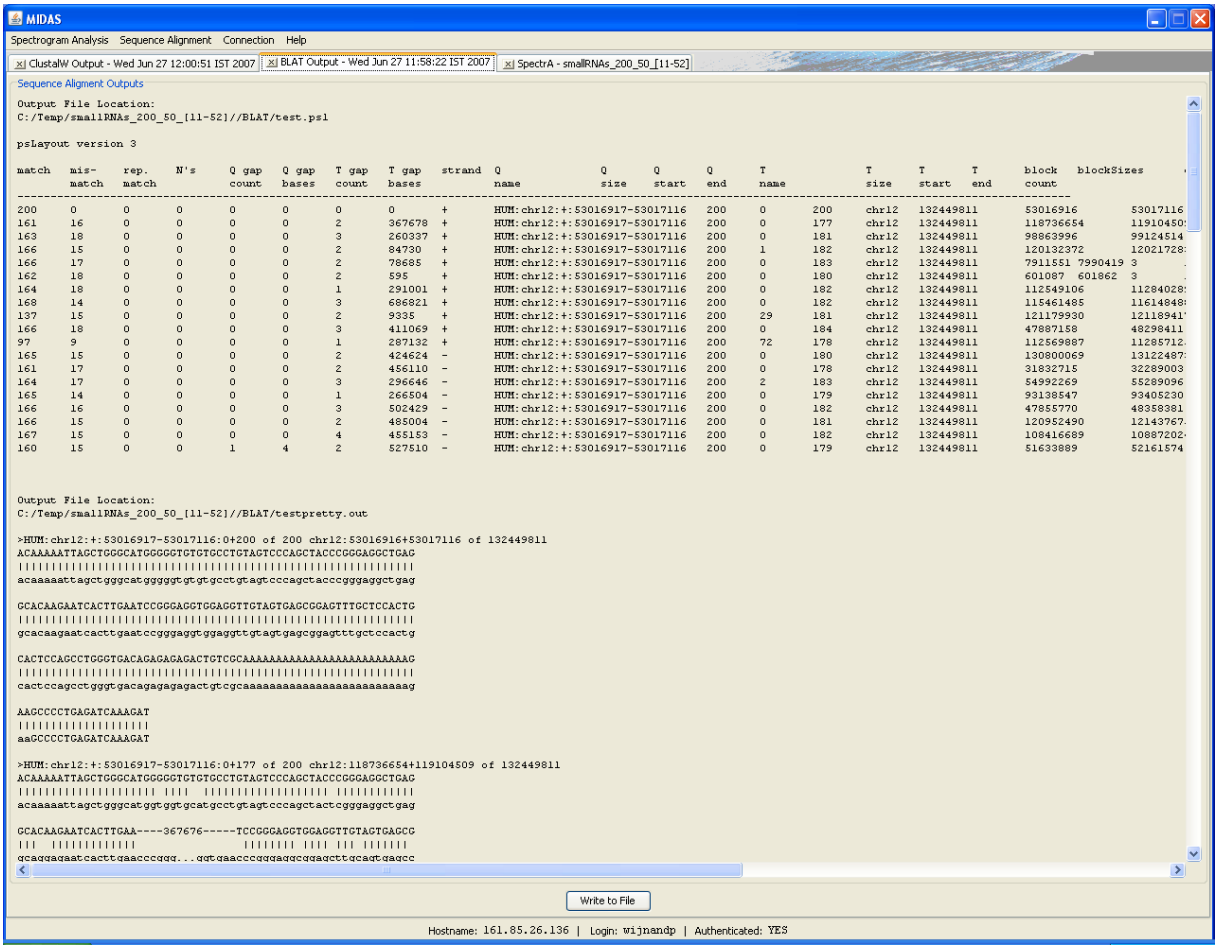


Figure 47 BLAT output

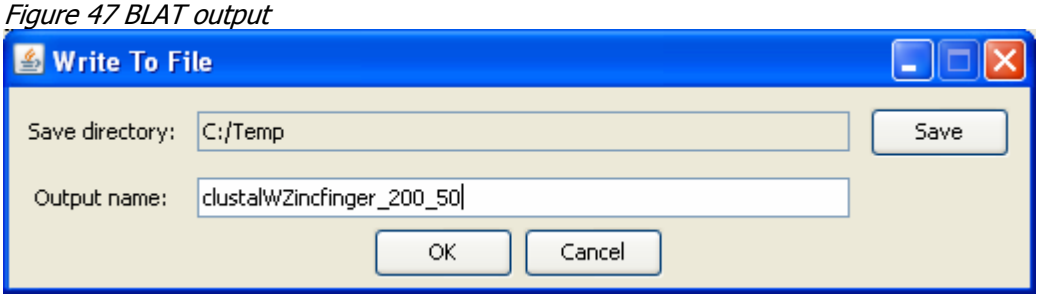


Figure 48 Write to file interface

The output of ClustalW is presented in Figure 49. On this panel the input given is also displayed. This is done by presenting the partial image and the annotations. The output of ClustalW is appended on the panel. It is also possible to write the output from ClustalW in a file on the desktop pc.

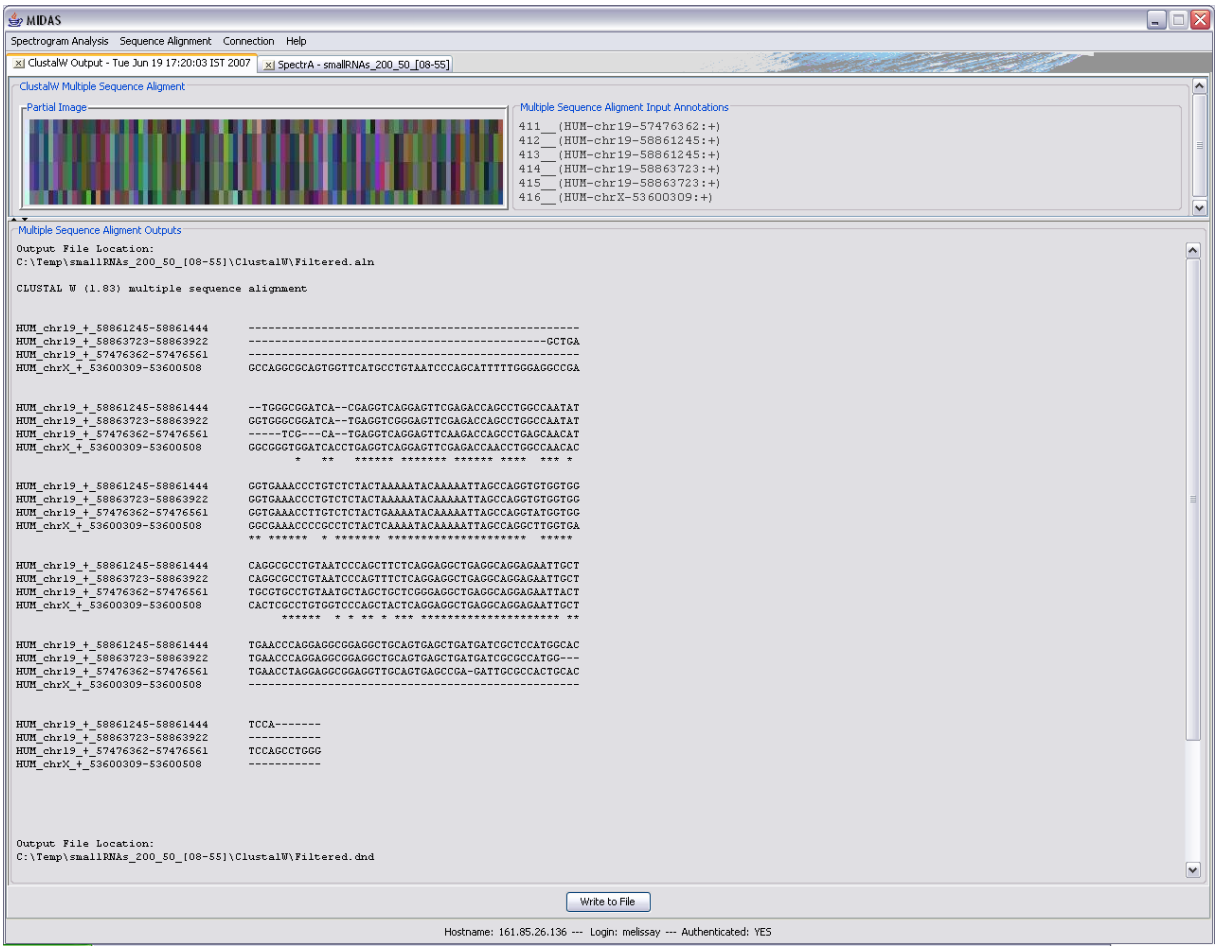


Figure 49 ClustalW output

It is also possible to load an external Spectrogram Analysis project into MIDAS for visualization (Figure 50). The project to be loaded has a constraint, that all the needed files to load visualization in MIDAS are present (see MIDAS constraints).

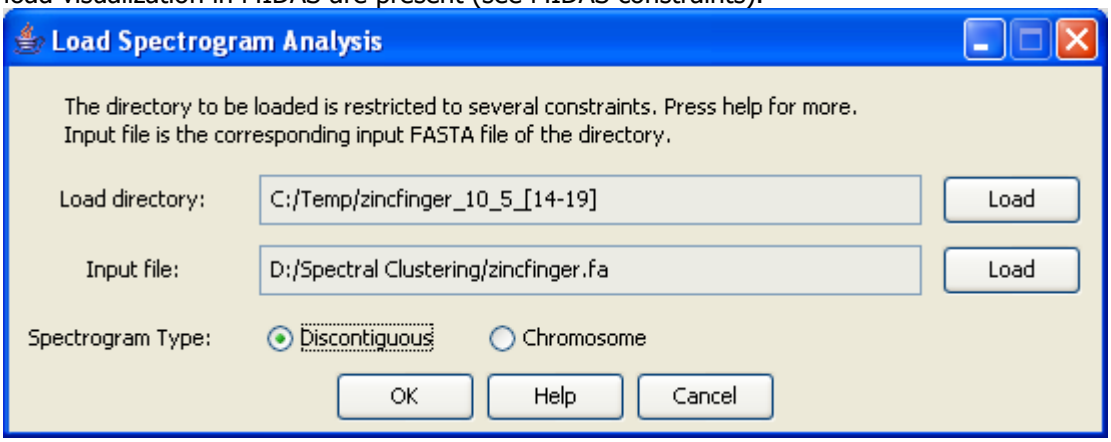


Figure 50 Load external spectrA project

Other features in MIDAS for Spectrogram Analysis are creating Discontiguous Spectrogram Video, Chromosome Spectrogram Video, and watch the videos.

sequence alignment can also be done on files which the users input. This can be selected in the menu bar. Sequence alignment for both BLAT and ClustalW are available.

Connecting to and disconnecting from the Philips Server are options given in the menu bar.

Help files are available in under the help menu item. Help files for BLAT, ClustalW and MIDAS can be opened. Also a stack trace of MIDAS can be viewed.

To summarize all the features of MIDAS:

- Run Discontiguous Spectrogram Analysis on user inputs
- Run Chromosome Spectrogram Analysis on user inputs
- Load a external Spectrogram Analysis
- Create Discontiguous Spectrogram Video
- Create Chromosome Spectrogram Video
- Watch Spectrogram Video
- Run sequence alignment (BLAT) on the output of a discontiguous Spectrogram Analysis
- Run multiple sequence alignment (ClustalW) on the output of a discontiguous Spectrogram Analysis
- Run sequence alignment (BLAT) on users input
- Run multiple sequence alignment (ClustalW) on users input
- Write output of BLAT to a file
- Write output of ClustalW to file
- Connect to Philips Server
- Disconnect from Philips Server
- View help file of BLAT
- View help file of ClustalW
- View help file of MIDAS
- View stack trace of processes of MIDAS

9 Future development

A lot of new ideas, improvements and extensions came up during the evaluation phase of this project. Due to time it was not possible to implement all these feedback. Since MIDAS 1.0 is the start of a new way to analyze DNA, there are a lot of possibilities for MIDAS to grow.

To understand these possibilities and their priority it is wise to summarize these in a MoSCoW analysis which separates the Must haves from the Should haves and the Could haves from the Wishful thinking requirements. Table 2 displays the MoSCoW analysis for MIDAS 2.0.

1. MIDAS Requirements	Description	Must Have	Should Have	Could Have	Wishful thinking
Spectrogram Analysis					
Load input	Load a FA-file as input for the Spectrogram Analysis	•			
Visualize Spectrogram Analysis	Obtain output of Spectrogram Analysis: the spectrogram video and the	•			
Visualize Clustering	Obtain an overview of all the present clustering in the analyzed sequence	•			
Run analysis without clustering	Run the spectrogram analysis without clustering but give the possibility for annotation	•			
Run S.A. with annotation	Run the spectrogram analysis with full annotation	•			
Run S.A. without annotation	Run the spectrogram analysis without full annotation	•			
Connect to Philips server					
Login	Give hostname, login name and password	•			
Connect	Connect to the server	•			
Disconnect	Disconnect to the server	•			
Connection world wide	Connect in a secure way to the Philips server at PIC Bangalore from everywhere in the world				•
Connection with pre-known use	Build an interface from which known users can select their name and save passwords for connection				•
BLAT					
Load input	Load input which the bioinformaticus selects in the annotation list from the Spectrogram Analysis	•			
Show output	Obtain the output of the BLAT sequence alignment	•			
Check input	Check inputted FA-file for correctness			•	
Run Blat separate	Run BLAT on user specified input file without running S.A. before	•			
ClustalW					
Load input	Load input which the bioinformaticus declares by giving a start line and an end line	•			
Show output	Obtain the output of the ClustalW sequence alignment	•			
Check input	Check inputted FA-file for correctness			•	
Run ClustalW separate	Run ClustalW on user specified input file without running S.A. before	•			
P-value score	Display a graph of p-value score, the significancy of the alignment		•		
Main interface					
Show background	Create a background for MIDAS to decorate the design			•	
Scale images horizontal	When images become to big for the interface, scale them horizontically	•			
System					
Failure handling	Saved data will not be lost	•			
System is fast	The code must be optimized for speed	•			
Deployabe	Create a deployable version of MIDAS	•			
Memory check	Check memory before running a project and give warning when there is not enough memory			•	
Reliable	System output cannot have errors, statistics must be correct	•			
Save project	Possibility to save your project			•	
Feedback					
Clickable cluster tree	Make the cluster tree clickable so that the user can interactivily select a cluster				•
Use correct names in panels	Make sure that all the names are biological correct	•			
SpectroVideo name	The video name should be the same as the input file with the corresponding windowsize and overlap	•			
Default database file BLAT	Set a default database by running BLAT based on the selection from the user		•		
Window based menu	Build up the menu like windows: use new, open, save, save as, etc.				•
Ranges of parameters in panel	Show the ranges of the parameters in the panel so that the bioinformaticus knows what to do			•	
Back button for changes	Create a back button to reset project and run it again with new settings			•	
Split BLAT and ClustalW	Create separate panels for BLAT and ClustalW	•			
Output folder with correct name	Output folder should contain the windowsize, overlap and inputfile name	•			

Table 3 MoSCoW analysis

10 Project Reflection

After completing MIDAS we can reflect our project and summarize our experiences. This was actually our first chance to put our theoretical knowledge into practice. Working at a multi-international like Philips is certainly not the same as studying at the university.

At the university you normally know exactly what to do, when the deadline is, what the assignment is and which persons have the answers to your questions during your assignment. In a company, these are the things you have to find out by yourself. We experienced that at a company all the employees have their tasks and their responsibilities and even more important: have their own schedules. Each question, meeting or appointment can be rescheduled since everybody is very busy.

We noticed that during our work in a company, we also had to deal with costs. For example production costs. To complete MIDAS we used a set of software tools and some of them were expensive. Then you start looking at other possibilities which are not expensive. We learned how to make decisions about software purchasing depending on the costs, possibilities, necessity and learning curve compared to the time we had.

At the Research department of PIC Bangalore we learned that it is very important not to lose preliminary work. Since many students will come and go their work has to be well documented so that it can be continued by other employees or new trainees. We documented MIDAS in a way that a new programmer does not have re-invent the wheel.

Being a computer science student does not mean that you are familiar with the bioinformatics field. During the development of MIDAS we created an interface for bioinformatics which has to fit to their needs. We have investigated this field for three months and saw some interesting possibilities to combine computer science with biological knowledge.

Being a trainee in India certainly brings some other challenges unrelated to work and education. These are challenges as differences in culture, food, hygiene, music and weather conditions. We learned that India is a country which has a lot of extremes: warm and cold, spicy and sweet, rich and poor, dry and rainy, loud or very silent.

Finishing your Bachelor with a Bachelor Project at Philips Bangalore in India certainly presented us a lot of challenges. After completing Project MIDAS during a three months internship we can look back to our work at Philips and our stay in India as something which broadened our view on the world, further educated us in the world of computer science and bioinformatics, gave us the opportunity to put our pre-training into practice and gave us good memories which will last for a life time.

11 Conclusions and recommendations

MIDAS has become a tool which combines Spectrogram Analysis with sequence alignment tools. By using MIDAS the bioinformatician is able to analyze a spectrogram together with the genomic annotation and the spectral clustering which can be aligned with the alignment tools BLAT or ClustalW. MIDAS is a standalone executable which can be used on single desktops. This all combined makes analyzing DNA sequences more efficient, less time consuming and less expensive.

MIDAS brings the bioinformatician efficiency in their work because they can align sequences which they find in a spectrogram all together in one application. Before MIDAS all these tasks had to be done manually. MIDAS erases 2/3 of all the tasks that were involved in the earlier situation.

Before project MIDAS was completed, the bioinformatician needed to run separate tools, and at most one at the time. MIDAS eliminates these tasks. MIDAS makes sure that all the tasks which do not have to be done by the user are done automatically and that user tasks are simplified. Analyzing DNA with MIDAS is not a time consuming job.

Since MIDAS is a deployed standalone executable which runs independently from developers' software, no expensive licences are needed to use MIDAS. The most expensive licence in the process of analyzing DNA usually is the MATLAB licence. MIDAS does not need this licensed software because it is compiled into a standalone.

There was a strong collaboration between the developers of MIDAS and the bioinformatician, MIDAS could be designed to fit the bioinformatician's needs.

During the development of MIDAS there have been performed several tests for the code and for the interface. A usability test is done and the feedback is used to redesign and improve the system. The usability test resulted very positive, which confirmed that the bioinformatician enjoyed using MIDAS.

To make sure that MIDAS can be used by everyone and even be extended by programmers, there are two manuals written: the Programmers Manual and the User Manual.

In the last evaluation phase of this project and during the end presentation for the research group of PIC Bangalore, a lot of new ideas for extensions came up. To summarize these new requirements and investigations on their priorities the MoSCoW analysis represents the new list of requirements for MIDASv2.0. We recommend that MIDAS will be extended to give the bioinformatician more tools, information and statistics to analyze DNA. MIDASv1.0 is only the beginning!

12 Literature

Scientific papers

N. Dimitrova, Y.H Cheung, M.Q. Zhang, Analysis and Visualization of DNA Spectrograms: Open Possibilities for the Bioinformatics Research, ACM MM 2006, Oct 25, 2006, pp. 1017 - 1024

N. Dimitrova, E. Santo, Improvement of spectral analysis as a genomic analysis tool

Jaques Cohen, Bioinformatics – An Introduction for Computer Scientists, Brandeis University, MA 02454

MATLAB tutorials

These can be found on the website of Mathworks (http://www.mathworks.com)
MATLAB Builder for Java 1.0
MATLAB Compiler 4.0
MATLAB Statistics Toolbox

Other resources

Wikipedia (http://www.wikipedia.com)
Java API (http://java.sun.com/j2se/1.5.0/docs/api/)

Appendix A: License Ganymed SSH-2

```
Copyright (c) 2005 - 2006 Swiss Federal Institute of Technology (ETH
Zurich),
  Department of Computer Science (http://www.inf.ethz.ch),
  Christian Plattner. All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

a.) Redistributions of source code must retain the above copyright
    notice, this list of conditions and the following disclaimer.
b.) Redistributions in binary form must reproduce the above copyright
    notice, this list of conditions and the following disclaimer in the
    documentation and/or other materials provided with the distribution.
c.) Neither the name of ETH Zurich nor the names of its contributors
may
    be used to endorse or promote products derived from this software
    without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS
BE
LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR
BUSINESS
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF
THE
POSSIBILITY OF SUCH DAMAGE.


The Java implementations of the AES, Blowfish and 3DES ciphers have
been
taken (and slightly modified) from the cryptography package released by
"The Legion Of The Bouncy Castle".

Their license states the following:

Copyright (c) 2000 - 2004 The Legion Of The Bouncy Castle
(http://www.bouncycastle.org)

Permission is hereby granted, free of charge, to any person obtaining a
copy
of this software and associated documentation files (the "Software"),
to deal
in the Software without restriction, including without limitation the
rights
```

```
to use, copy, modify, merge, publish, distribute, sublicense, and/or
sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included
in
all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS
OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL
THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING
FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN
THE SOFTWARE.
```

Appendix B: License FAT-jar

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and
(2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we
want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free

program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under

these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component

itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed

through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF

MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS
TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING
WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES,
INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING
OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED
TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY
YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER
PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
    Gnomovision version 69, Copyright (C) year name of author
    Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type
`show w'.
    This is free software, and you are welcome to redistribute it
    under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate
parts of the General Public License. Of course, the commands you use may
be called something other than `show w' and `show c'; they could even be
mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
  Yoyodyne, Inc., hereby disclaims all copyright interest in the
program
  `Gnomovision' (which makes passes at compilers) written by James
Hacker.

  <signature of Ty Coon>, 1 April 1989
  Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into
proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the
library. If this is what you want to do, use the GNU Library General Public License instead of this License.

Appendix C: Test Cases

Test Case: Input Chromosome for analysis

Actions performed:

- Checked pre-conditions
- Select chromosome analysis
- Set input file path
- Set annotation file path
- Set output path
- Set start position
- Set orientation
- Set clustering flag
- Set annotation flag
- Set windows size
- Set window overlap
- Test some error input values to see what error message is given to the user
- Pressed ok
- Checked post-conditions

Problems: Tree was not visualized but no error was given by our console. The image was too large and did not fit on the screen.

Cause: The input file contained 67 sequences. On one screen you can visualize 50 sequences max. To check whether an input file is bigger than 50, we divide the number of sequences by the max number of sequences which can be displayed. If your answer is less than or equal to one, you will have one screen with a max of 50 sequences. If your answer is more than one, you split the images into separate frames of 50 sequences each. In this case, 67/50 should be more than one, but Java casts this answer to an Integer which means that the answer was exactly one. The visualization of the cluster tree went wrong because MIDAS saw only one image which was too big, and needed two images of the right size.

Solution: We put a cast to double around the division in MATLAB, and this solved the problem.

Result: Problem solved

Remarks: After this problem we tested sequences which were less than 50 or more than 100 just to be sure that our adjustment in the code was working for all the cases. After these tests, everything seemed to work fine.

Test Case: Input Discontiguous DNA for analysis

Actions performed:

- Checked pre-conditions
- Select discontiguous analysis
- Set input file path
- Set annotation dir path
- Set output path
- Set windows size
- Set window overlap

- Test some error input values to see what error message is given to the user
- Pressed ok
- Checked post-conditions

Problems: Tree was not visualized but no error was given by our console. The image was too large and did not fit on the screen.

Cause: The input file contained 67 sequences. On one screen you can visualize 50 sequences max. To check whether an input file is bigger than 50, we divide the number of sequences by the max number of sequences which can be displayed. If your answer is less than or equal to one, you will have one screen with a max of 50 sequences. If your answer is more than one, you split the images into separate frames of 50 sequences each. In this case, 67/50 should be more than one, but Java casts this answer to an Integer which means that the answer was exactly one. The visualization of the cluster tree went wrong because MIDAS saw only one image which was too big, and needed two images of the right size.

Solution: We put a cast to double around the division in MATLAB, and this solved the problem.

Result: Problem solved

Remarks: After this problem we tested sequences which were less than 50 or more than 100 just to be sure that our adjustment in the code was working for all the cases. After these tests, everything seemed to work fine.

Test Case: Use Spectrogram Analysis

Actions performed:

- Checked pre-conditions
- Checked Console output
- Checked spectrogram images and annotations
- Checked post-conditions

Problems: No problems

Cause: -

Solution: -

Result: *Use Spectrogram Analysis* is tested and no errors were found

Remarks: -

Test Case: Create Spectrogram Video

Actions performed:

- Checked pre-conditions
- Select create Spectrogram video option from menu for a chromosome or a discontiguous chromosome.
- Inspect the video during its creation
- Checked post-conditions

Problems: When a video is manually erased from the folder, the system still 'thinks' that the video is created and gives a message that the video exists.

Cause: The system only checked if the video folder was present instead of checking the video file

Solution: The system currently checks if there is any file in the video folder which has an extension .avi.

Result: After solving the problem, the next test for *Create Spectrogram Video* resulted without errors.

Remarks: A few more tests have been executed to check what happens if there is an avi-file, if there is no avi-file, if the bioinformaticus clears the folder and even when the bioinformaticus deletes the folder. Every test resulted positively.

Test Case: Watch Spectrogram Video

Actions performed:

- Checked pre-conditions
- Checked Console output
- Checked Spectrogram Video
- Checked post-conditions

Problems: No problems

Cause: -

Solution: -

Result: *Watch Spectrogram Video* is tested and no errors were found

Remarks: -

Test Case: Overview all present clustering DNA

Actions performed:

- Checked pre-conditions
- Checked Console output
- Checked clustering
- Checked post-conditions

Problems: No problems

Cause: -

Solution: -

Result: *Overview all present clustering DNA* is tested and no errors were found

Remarks: -

Test Case: Login to Philips server

Actions performed:

- Checked pre-conditions
- Selected a sequence for input for BLAT and in the second test for ClustalW
- Clicked on run BLAT and in the second test on run ClustalW
- Filled in hostname
- Filled in login name
- Filled in password
- Pushed OK
- Checked authenticated message
- Checked post-conditions

Problems: When a wrong input was given to ClustalW, an error message was given to the user. By clicking on the OK button of the error message, login was started. Login should not start if a wrong input is given.

Cause: The "if" condition in the code used OR instead of AND so there was a bug in the interface.

Solution: We repaired the condition

Result: Problem solved

Remarks: After solving this problem, each part of the system were login was needed was tested, and in the end every test resulted without errors.

Test Case: Use BLAT

Actions performed:

- Checked pre-conditions
- Set the normal settings and additional settings
- Click on OK button
- Checked output BLAT
- Checked Console output
- Checked post-conditions

Problems: No problems

Cause: -

Solution: -

Result: *Use BLAT* is tested and no errors were found

Remarks: -

Test Case: Use ClustalW

Actions performed:

- Checked pre-conditions
- Set the normal settings and additional settings
- Click on OK button
- Checked output ClustalW
- Checked Console output

- Checked post-conditions

Problems: No problems

Cause: -

Solution: -

Result: *Use ClustalW* is tested and no errors were found

Remarks: -

Test Case: Load existing project
Actions performed:

- Checked pre-conditions
- Selected the folder where the output files are located from the existing project
- Select the input file which is used to create the existing project we want to display
- Checked Console output
- Checked post-conditions

Problems: No problems

Cause: -
Solution: -
Result: *Load existing project* is tested and no errors were found

Remarks: -

Test Case: Run BLAT without existing project
Actions performed:

- Checked pre-conditions
- Set the normal settings and additional settings
- Click on OK button
- Checked output BLAT
- Checked Console output
- Checked post-conditions

Problems: No problems

Cause: -

Solution: -

Result: *Run BLAT* is tested and no errors were found

Remarks: -

Test Case: Run ClustalW without existing project
Actions performed:

- Checked pre-conditions
- Set the normal settings and additional settings
- Click on OK button
- Checked output ClustalW
- Checked Console output
- Checked post-conditions

Problems: No problems

Cause: -

Solution: -

Result: *Run ClustalW* is tested and no errors were found

Remarks: -

Test Case: Disconnect from Philips Server

Actions performed:

- Checked pre-conditions
- Click on the option 'Disconnect from server' from the menu
- Check the popup given by the system which explains that the system is disconnected
- Checked post-conditions

Problems: No problems

Cause: -

Solution: -

Result: *Disconnect to Philips Server* is tested and no errors were found

Remarks: -

Appendix D: User Manual

MIDAS v1.0

Multimodal Interface for DNA Alignment of Sequences

Melissa Yuen Shan Cheung
Software Technology, Computer Science
Technical University of Delft, The Netherlands
myscheung@gmail.com
Wijnand Paul van den Haak
Media and Knowledge Technology, Computer Science
Technical University of Delft, The Netherlands
paulvandenhaak@gmail.com

April 2007 – June 2007

Table of contents

1. System description

Major research efforts in Bioinformatics include sequence alignment, gene finding, genome assembly, protein structure alignment, protein structure prediction, prediction of gene expression and protein-protein interactions, and the modeling of evolution. To perform these specific tasks different tools are used. Using these tools separately is a time consuming, inefficient and expensive process.

MIDAS is a tool that integrates sequence alignment, genome annotation, and spectral clustering and alignment under the same application. The DNA data is first transformed into Fourier domain and clustered in MATLAB based on Euclidean distances between the sequences. This tool allows visualizing the DNA spectra together with a hierarchical tree in a multimodal interface. This in turn enables a bioinformatician to analyze patterns of a group of sequences. Visualisations of other clustering or sorting techniques are also possible by using this technique.

MIDAS is a standalone application which provides an interface around standard sequence alignment tools such as BLAT, ClustalW, as well as newer alignment tools such as Spectrogram analysis via integrating MATLAB code, server connections and data visualizations

2. Installation

Before you can start working with MIDAS, the application must be installed. MIDAS contains an installer which installs the program for you.
These next steps will guide you through the installation process.

1. Check your Windows settings. MIDAS runs optimal on Windows XP, with resolution 1280x1024.
2. Run setup_MIDASv02.exe.
3. Click *Install*.
4. Follow the instructions during the installation,
   NOTE: .NET environment is not needed, just press OK.
   JRE 1.5 (Java Runtime Environment) must be installed. If the JRE is already installed on your computer, the installer will inform you and skip the JRE installation
   Indeo5 codec is needed so you have to follow the steps in the installation of this installer. You do not have to follow these steps if you have already installed this codec on your pc.
5. The output directory of MIDAS is default set to C:\MIDAS.
6. After the installation push *done*
7. Run MIDASv1.0.jar in C:\MIDAS
8. MIDAS starts.

3. Spectrogram Analysis

MIDAS can be used to run a Discontiguous Spectrogram Analysis or a Chromosome Spectrogram Analysis. These are two different actions. In the first paragraph is explained how to run a Discontiguous Spectrogram Analysis. The second paragraph explains the same for a Chromosome Spectrogram Analysis. The output is explained in the third paragraph and how to create and run a spectrogram video on this output is explained in the fourth paragraph. In the last paragraph is explained how to load an existing project.

3.1 Run Discontiguous Spectrogram Analysis

These steps will guide you how to run Discontiguous Spectrogram Analysis.

1. In the menu bar, select *Spectrogram Analysis*.
2. Select *Run New Spectrogram Analysis*, a new spectrogram settings window (figure 51) will appear.

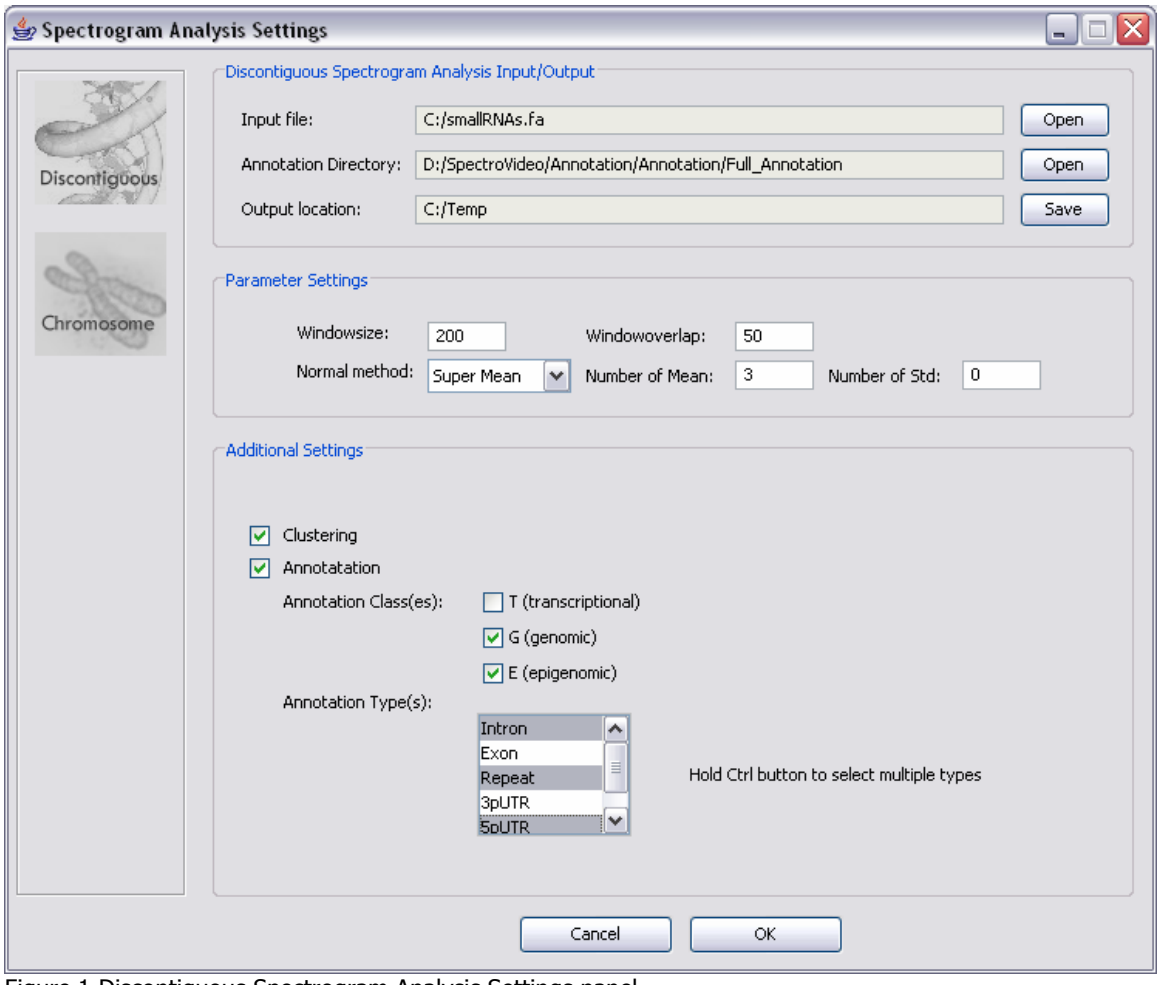


Figure 1 Discontiguous Spectrogram Analysis Settings panel

3. Select *Discontiguous*, the default panel is the discontiguous spectrogram settings panel.
4. Give the correct input arguments on the settings panel.
    a. Input file: Press *Open* to select an input FASTA file.
    b. Annotation Directory: Press *Open* to select an annotation directory.
    c. Output Location: Press *Save* to select a directory for saving your output.
    d. WindowSize: window size should be an even number, between 0 and 50000.

e. WindowOverlap: window overlap is at maximum windowSize – 1.
f. Normal method: select an option in the dropdown list.
g. Number of Mean: must be a positive number.
h. Number of Std: must be a positive number.
i. Clustering: check for clustering, uncheck for without clustering.
j. Annotation: check for annotation, uncheck for without annotation.
    i. Annotation Classes: at least one should be checked.
    ii. Annotation Types: at least one should be selected.

NOTE: Example input (found_spectral_noalign.fa will run approximately 10 minutes with annotation)

5. Press *OK*, and wait for the console to appear. The console will show the processes MIDAS is executing.
6. The output will be loaded. See output panel for Spectrogram Analysis (2.3).

3.2 Run Chromosome Spectrogram Analysis

These steps will guide you how to run Chromosome Spectrogram Analysis.

1. In the menu bar, select *Spectrogram Analysis*.
2. Select *Run New Spectrogram Analysis*, a new spectrogram settings window (figure 2) will appear.

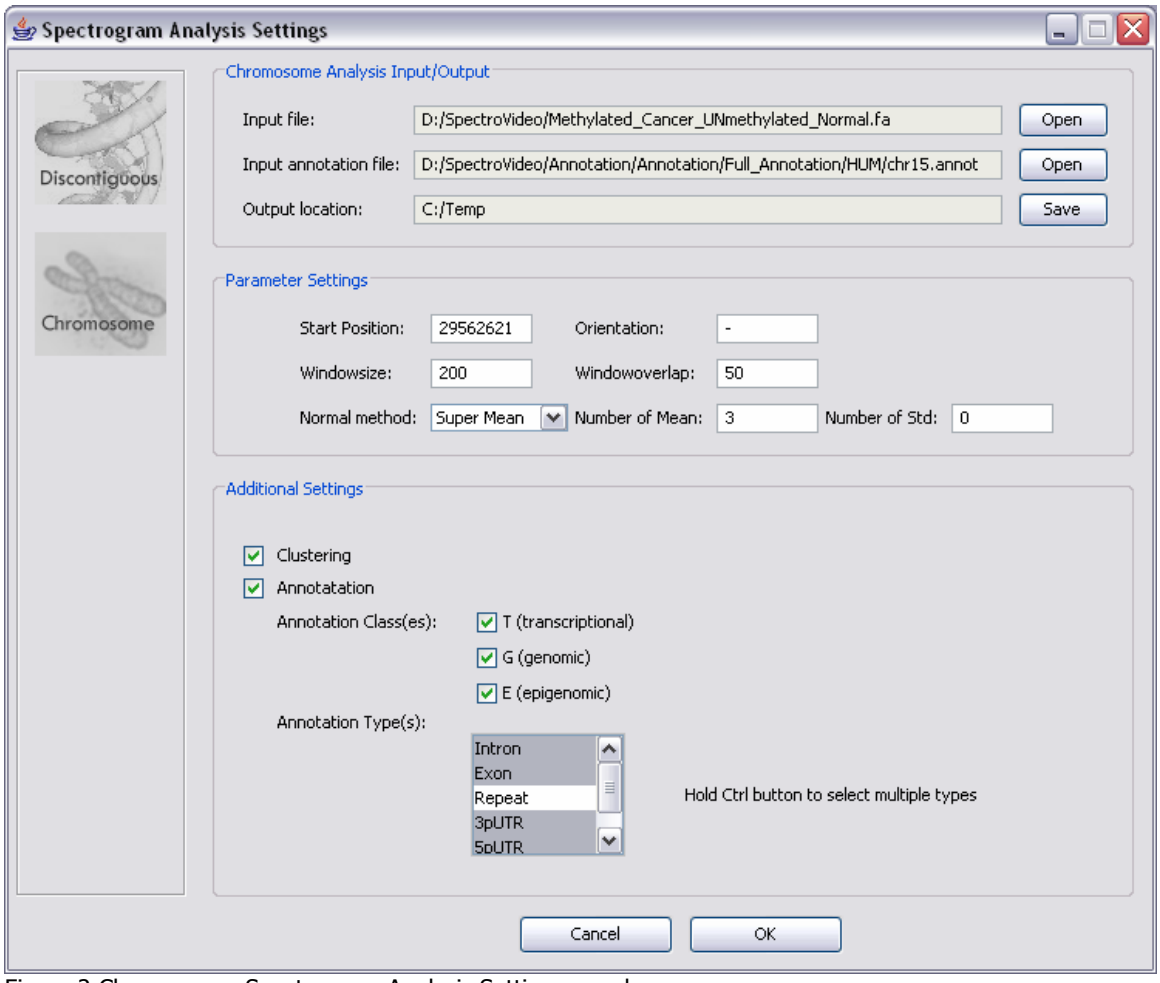


Figure 2 Chromosome Spectrogram Analysis Settings panel

3. Select *Chromosome*
4. Give the correct input arguments on the settings panel.
    a. Input file: Press *Open* to select an input FASTA file.
    b. Input annotation file: Press *Open* to select an annotation file.
    c. Output Location: Press *Save* to select a directory for saving your output.
    d. Start Position: must be a positive number.
    e. Orientation: either – or +.
    f. WindowSize: windowSize should be an even number, and between 0 and 50000.
    g. WindowOverlap: window overlap is at maximum windowSize – 1.
    h. Normal method: select an option in the dropdown list.
    i. Number of Mean: must be a positive number.
    j. Number of Std: must be a positive number.
    k. Clustering: check for clustering, uncheck for without clustering.
    l. Annotation: check for annotation, uncheck for without annotation.
        i. Annotation Classes: at least one should be checked.
        ii. Annotation Types: at least one should be selected.

    NOTE: Example input (Methylated_Cancer_UNmethylated_Normal.fa will run approximately 10 minutes with annotation)
5. Press OK, and wait for the console to appear. The console will show the processes MIDAS is executing.
6. The output will be loaded. See output panel for Spectrogram Analysis (2.3).

3.3 Output panel from Spectrogram Analysis

After you have completed a Spectrogram Analysis, you can analyze the output. MIDAS works with tabbed panels. The output panel for Spectrogram Analysis contains the content of the Annotation file, the Spectrogram Images, a clustered hierarchical tree and a slider. If the Spectrogram Analysis type corresponds to a Discontiguous Spectrogram Analysis you can push the *sequence alignment* button to make a BLAT/ClustalW panel visible (figure 3). This feature is not yet available for Chromosome Spectrogram Analysis so only the annotation, the spectrogram images and the cluster tree are visible here(figure 4).

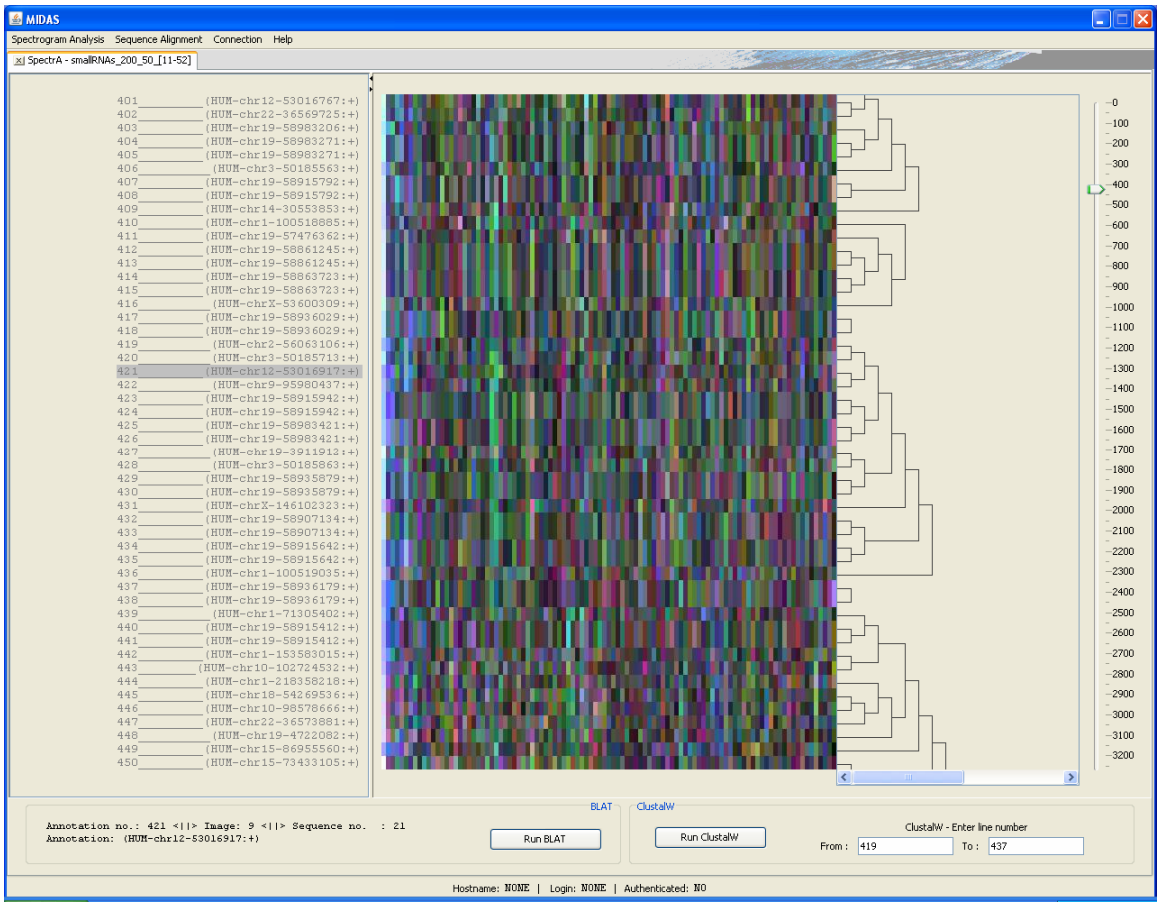


Figure 3 Output from a Discontiguous Spectrogram Analysis

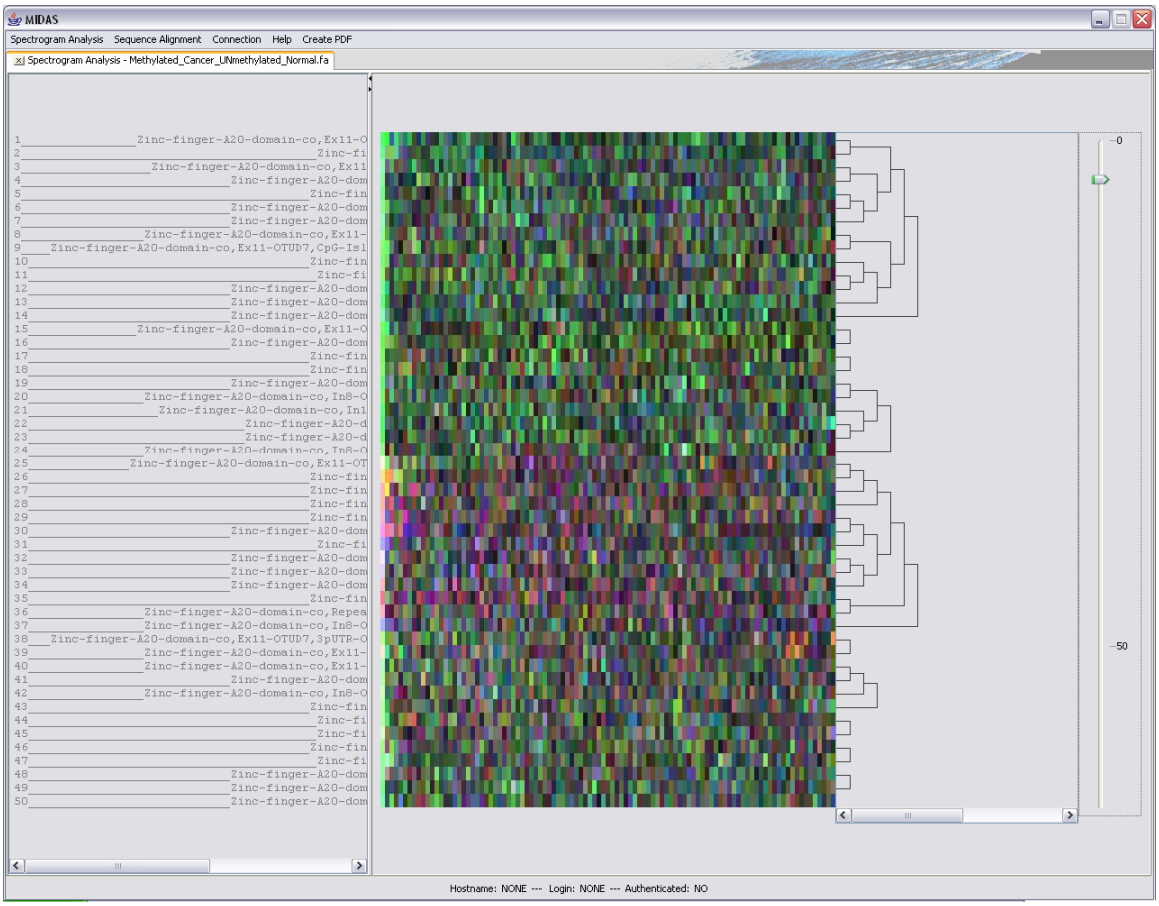


Figure 4 Output from a Chromosome Spectrogram Analysis

One Spectrogram Image contains 50 sequences. As the output panel shows one spectrogram image at the time, the corresponding 50 annotations and the cluster tree for that part are loaded. The slider can be used to "scroll" through all the Spectrogram Images. After each transition between images, the corresponding annotations and cluster tree are loaded real time. The labels of the slider are sequence numbers.

3.4 Run Spectrogram Video

The Spectrogram Video can be created in MIDAS for a Chromosome or Discontiguous Spectrogram Analysis output. The videos that are created or shown belong to the current active and visible tab of MIDAS.

Create Spectrogram Video

1. First a Spectrogram Analysis output panel/tab should be visible. (Either from *Run Spectrogram Analysis* or *Load Existing Project*)
2. In the menu bar, select *Spectrogram Analysis*, select *Create Spectrogram Video*, and select either *Create Discontiguous Spectrogram Video* or *Create Chromosome Spectrogram Video*, corresponding to the Spectrogram Analysis Type of the current tab.
3. The Spectrogram Video will be created in MATLAB. Do not interrupt the video, because during the creation of the video, MATLAB records the current desktop.

Watch Spectrogram Video

1. Select the Spectrogram Analysis output panel/tab from which the video should be played

2. In the menu bar, select *Spectrogram Analysis*, select *Open Video Current Tab.*
3. The video will be shown in Media Player if it exists. If the video does not exists an error message will pop up telling you to create the video first.

3.5 Load existing Spectrogram Analysis project

In MIDAS you can load an existing Spectrogram Analysis Project. This project is restricted that it should have the same format and files as it would have in a MIDAS Spectrogram Analysis output directory.

Load Spectrogram Analysis Project

1. In the menu bar, select *Spectrogram Analysis*, select *Load Existing Project.* The load window will pop up (figure 5).
2. Give the correct input arguments:
    a. Load directory: Press *Load* to select a directory.
    b. Input File: Press *Load* to select an input file.
    c. Spectrogram Type: Select correct type.
    NOTE: very important is to give correct input arguments, as MIDAS does not detect errors concerning content.

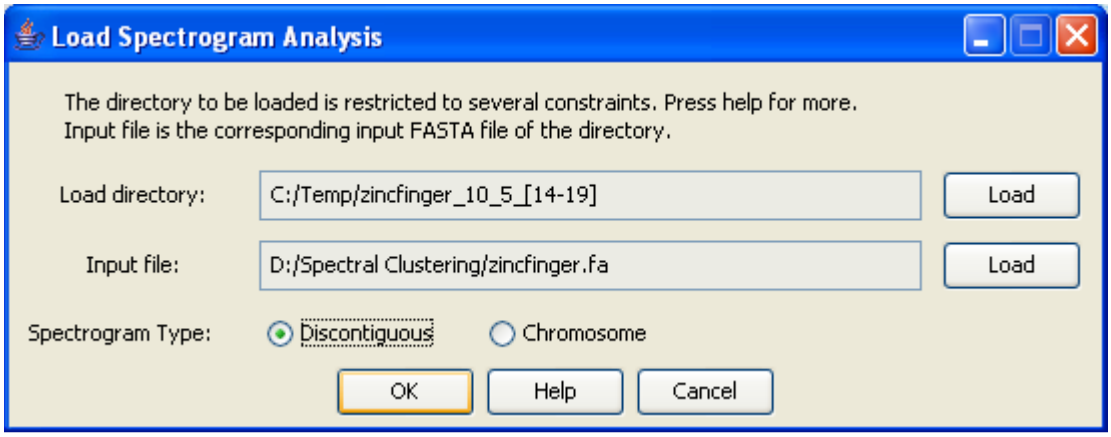


Figure 5 Load Spectrogram Analysis panel

4. sequence alignment

NOTE: This option is only available for a Discontiguous Spectrogram Analysis.

To perform a sequence alignment by running BLAT or ClustalW you must have a current project on an active tab in MIDAS. This can either be the Spectrogram Analysis you are currently working on or an existing project which you have to load first. On this tab in MIDAS you can run BLAT or ClustalW by following the process which is described in this paragraph.

4.1 Run BLAT

1. Press *sequence alignment* in order to start sequence alignment. Two buttons will appear, *Run BLAT* and *Run ClustalW*.
2. Select in the Annotation list an annotation for BLAT.
3. Press *Run BLAT*.
4. Login will appear if you are not logged on the server yet. You need to have an account for the Philips Linux Server.
   Note: The file transfer between the Server and local machine will cause MIDAS to freeze some time. Please be patient.
5. Give the correct input arguments:
   a. BLAT location: location of BLAT on the server. There is a possibility to *Edit* the location.
   b. Query: the selected query input. There is a possibility to *Show* the query in Notepad.
   c. Database: the database is on the Linux server. Give the correct database.
   d. Output file: set name and extension.
   e. Database Type: select the database type.
   f. Query Type: select the query type.
   g. Additional settings: give command as in a command shell, see BLAT help for more information on the options.

BLAT input example:

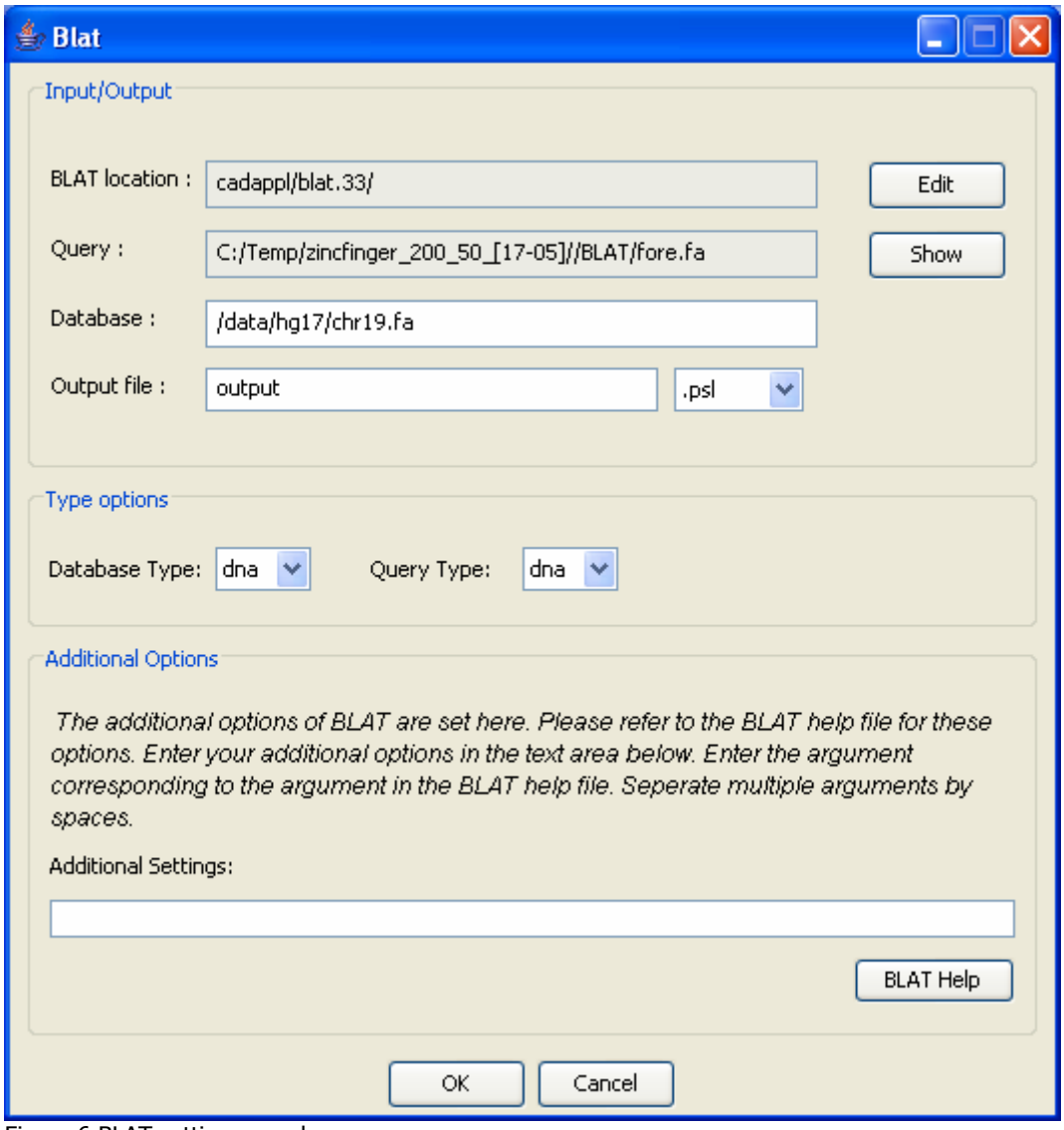


Figure 6 BLAT settings panel

6. Press OK button. BLAT will run.
   Note: The file transfer between the Server and local machine will cause MIDAS to freeze some time. Please be patient.
7. The output panel will be loaded with the output files (figure 7).
   Note: if some mismatch occurred with the input arguments, the output panel can be empty, as MIDAS does not detects errors concerning content.

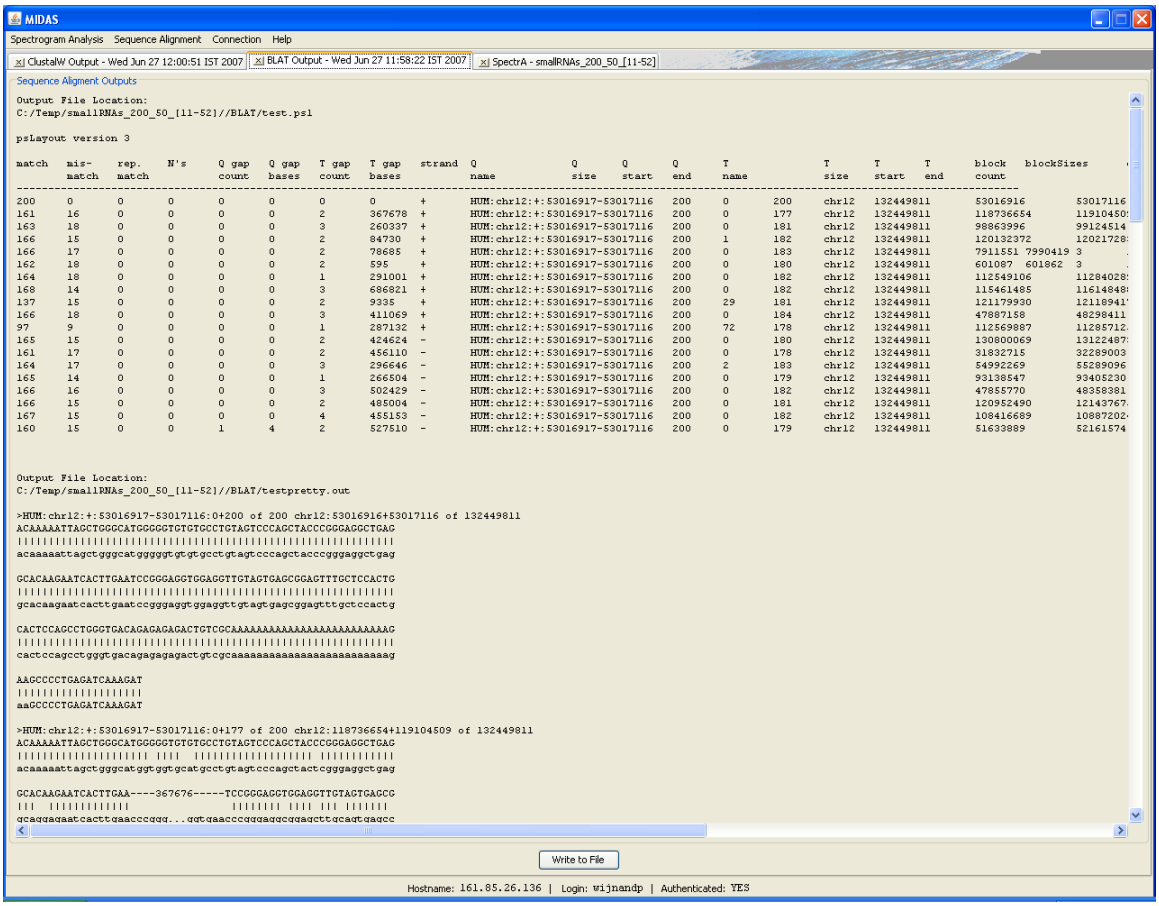


Figure 7 BLAT output panel

The figure displays the *Write to File* button which gives the possibility to write the BLAT output to a text file. After pushing this button, a save dialog will pop up (figure 8).

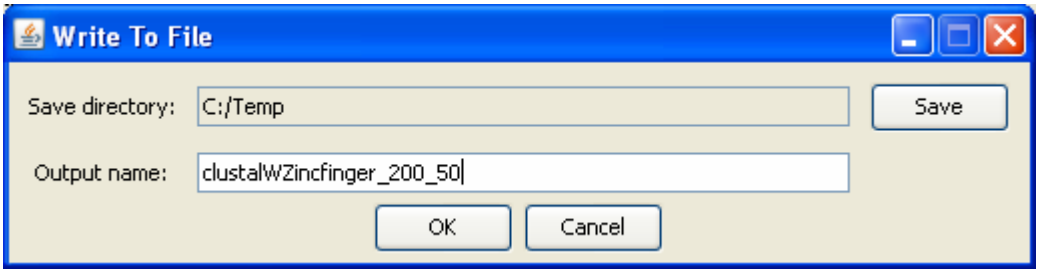


Figure 8 Write to File dialog

4.2 Run ClustalW

1. Press *sequence alignment* in order to start sequence alignment. Two buttons will appear, *Run BLAT* and *Run ClustalW*.
2. Give the range in the input fields for ClustalW of the sequences to align.
3. Press *Run ClustalW*.
4. Login will appear if you are not logged on the server yet. You need to have an account for the Philips Linux Server.
   Note: The file transfer between the Server and local machine will cause MIDAS to freeze some time. Please be patient.

5. Give the correct input arguments:
    a. ClustalW location: location of ClustalW on the server. There is a possibility to *Edit* the location.
    b. Query: the selected query input. There is a possibility to *Show* the query in Notepad.
    c. Output Directory: shows the output directory.
    d. GapOpen: number required for gap open penalty.
    e. Bootstrap: number of bootstraps.
    f. Additional settings: give command as in a command shell, see ClustalW help for more.

ClustalW input example:

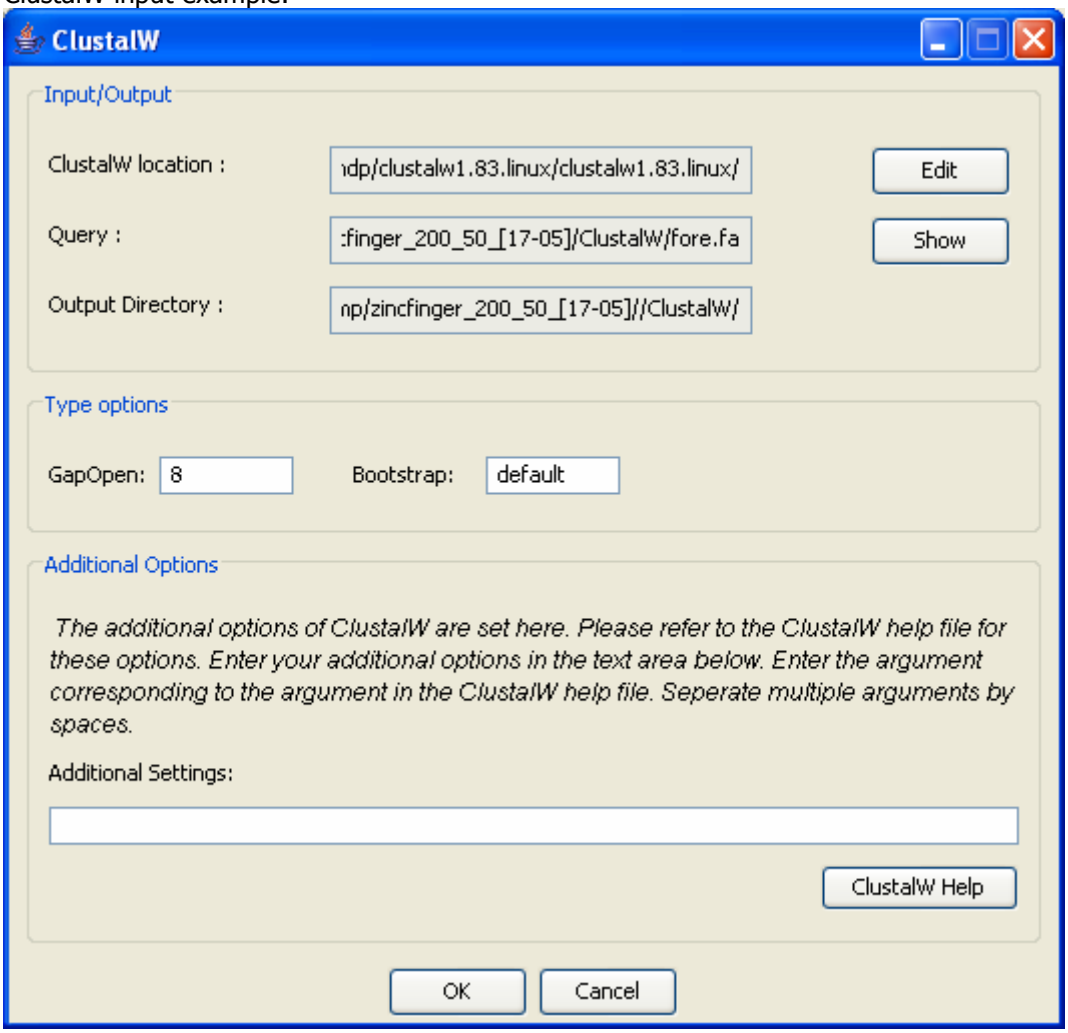


Figure 9 ClustalW settings panel

6. Press OK button. ClustalW will run.
   Note: The file transfer between the Server and local machine will cause MIDAS to freeze some time. Please be patient.
7. The output panel will be loaded with the output files (figure 10).
   Note: if some mismatch occurred with the input arguments, the output panel can be empty, as MIDAS does not detects errors concerning content.

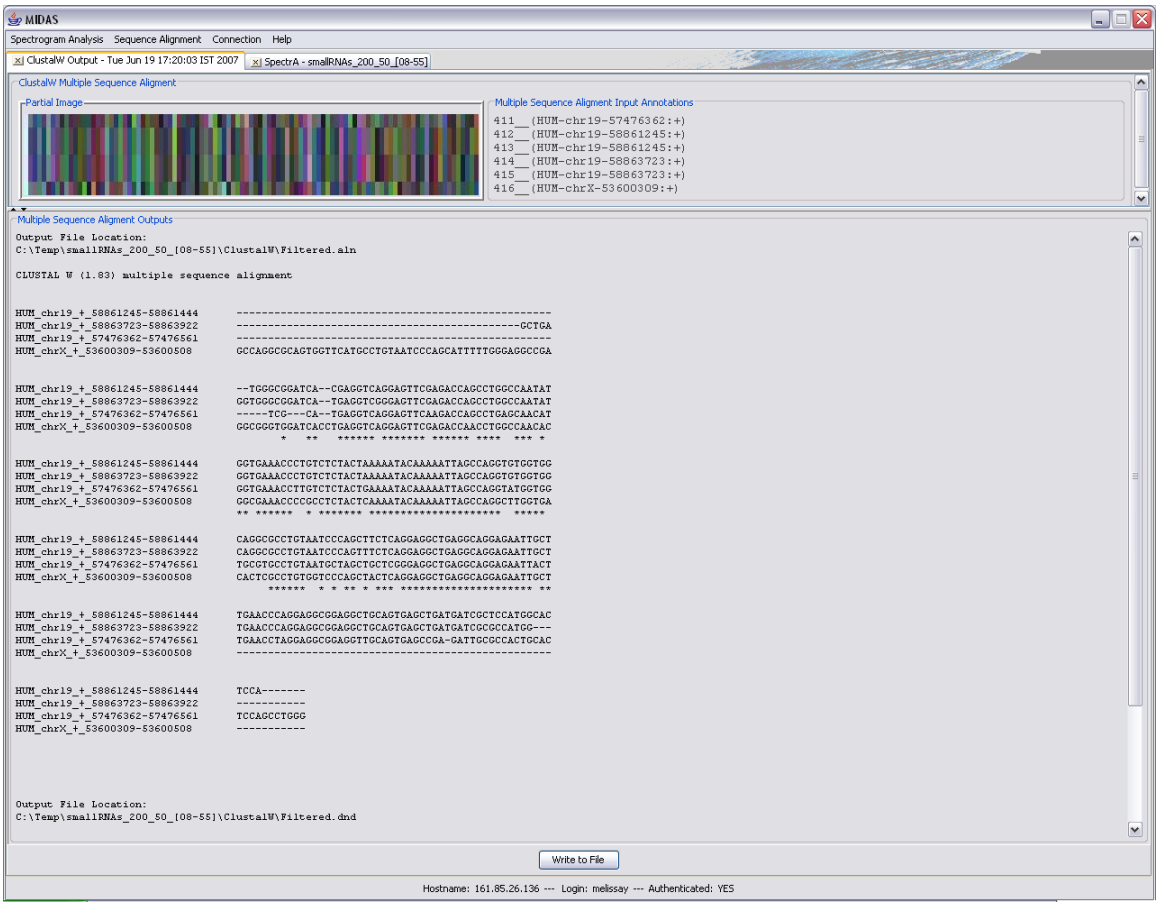


Figure 10 ClustalW settings panel

The figure displays the *Write to File* button which gives the possibility to write the ClustalW output to a text file. After pushing this button, a save dialog will pop up (figure 11).

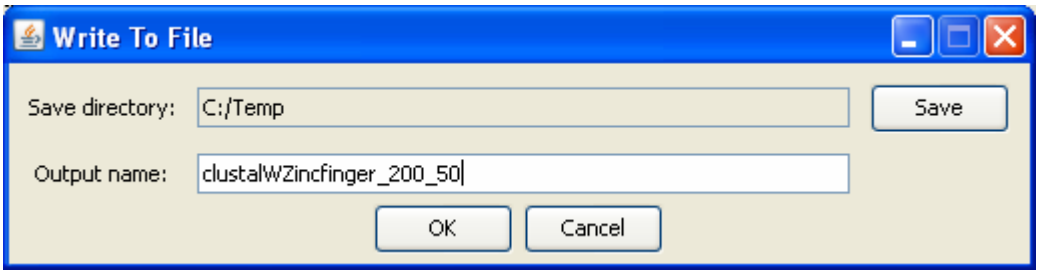


Figure 11 Write to File dialog

5 Server Connection

In order to run BLAT or ClustalW you must be connected to the Philips Linux server. The genome database and both BLAT and ClustalW are located on this server.

5.1 Connect

To connect to the Linux server, these steps must be completed:

1. Select *Connection* from the menu bar and choose *Connect to Server*. The login interface will pop up (figure 12)
2. Change the hostname if necessary
3. Fill in your login and your password, and press *OK*

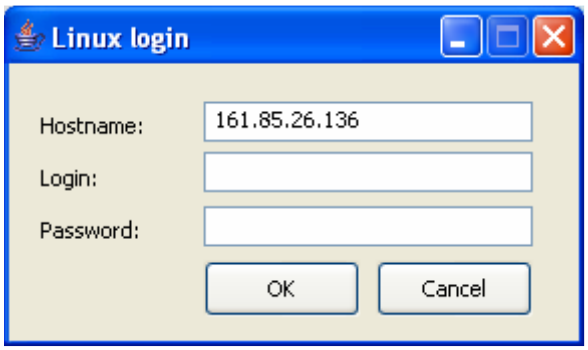


Figure 12 Connection panel

4. If you have typed everything right, you should be connected now.

5.2 Disconnect

To disconnect from the Linux server, only one action has to be performed. Select *Connection* from the menu bar and choose *Disconnect from Server*. You should be disconnected in a few seconds.